US 20020186656A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2002/0186656 A1**

Vu (43) **Pub. Date: Dec. 12, 2002**

(54) **AUTOMATIC LOAD BALANCING IN SWITCH FABRICS**

(76) Inventor: **Chuong D. Vu**, Simi Valley, CA (US)

Correspondence Address:
CHRISTIE, PARKER & HALE, LLP
350 WEST COLORADO BOULEVARD
SUITE 500
PASADENA, CA 91105 (US)

(21) Appl. No.: **10/020,491**

(22) Filed: **Dec. 11, 2001**

Related U.S. Application Data

(60) Provisional application No. 60/289,557, filed on May 7, 2001.

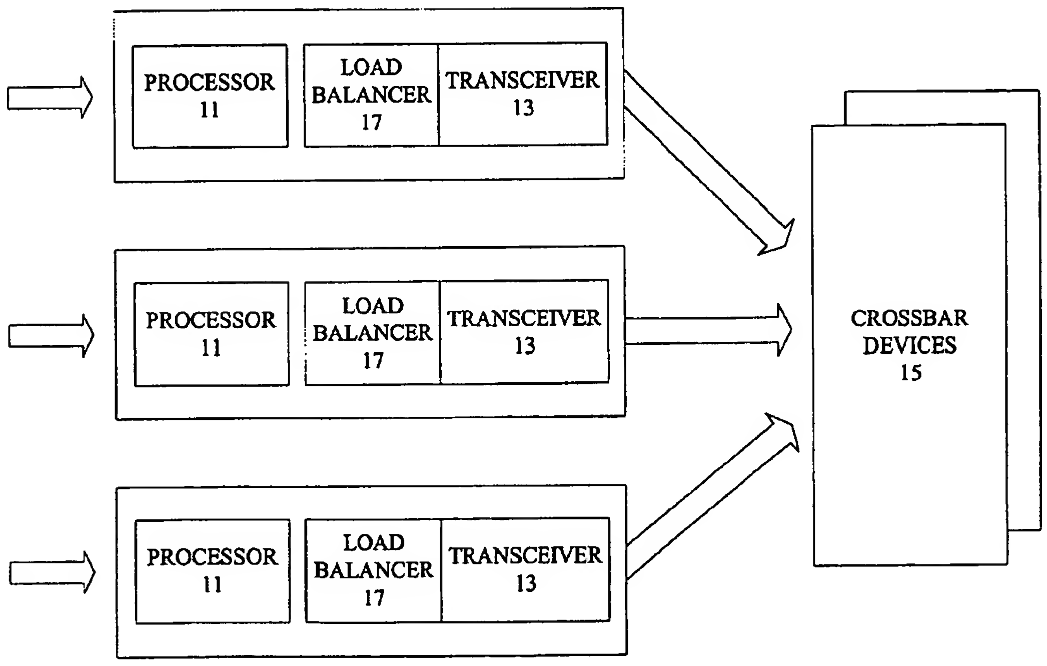
Publication Classification

(51) **Int. Cl.**^{7} **H04J 1/16**; H04J 3/14

(52) **U.S. Cl.** **370/229**; 370/412; 370/413; 370/415

(57) **ABSTRACT**

A load balancing system and method for network nodes is provided. The load balancing system includes crossbar devices, queues to receive data and a load balancer. The load balancer determines the amount of data in each of the queues and sends data to specific crossbar devices based on the amount of data in each queue. The queues include a high priority queue and a number of non-high priority queues.



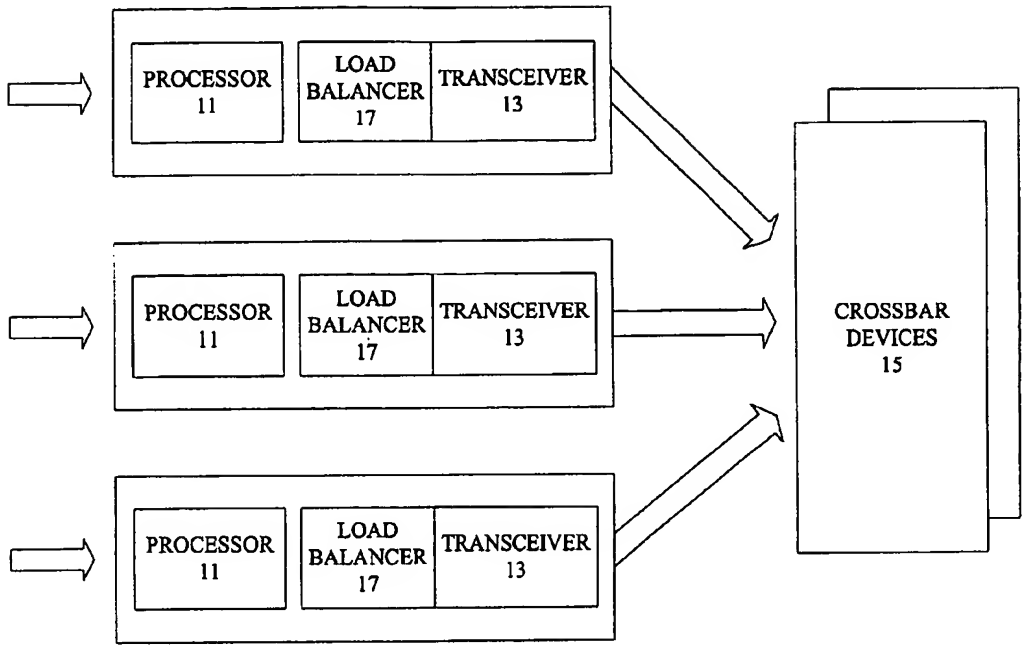

PROCESSOR
11
LOAD
BALANCER
17
TRANSCEIVER
13
PROCESSOR
11
LOAD
BALANCER
17
TRANSCEIVER
13
PROCESSOR
11
LOAD
BALANCER
17
TRANSCEIVER
13
CROSSBAR
DEVICES
15

FIG. 1

FIG. 2A

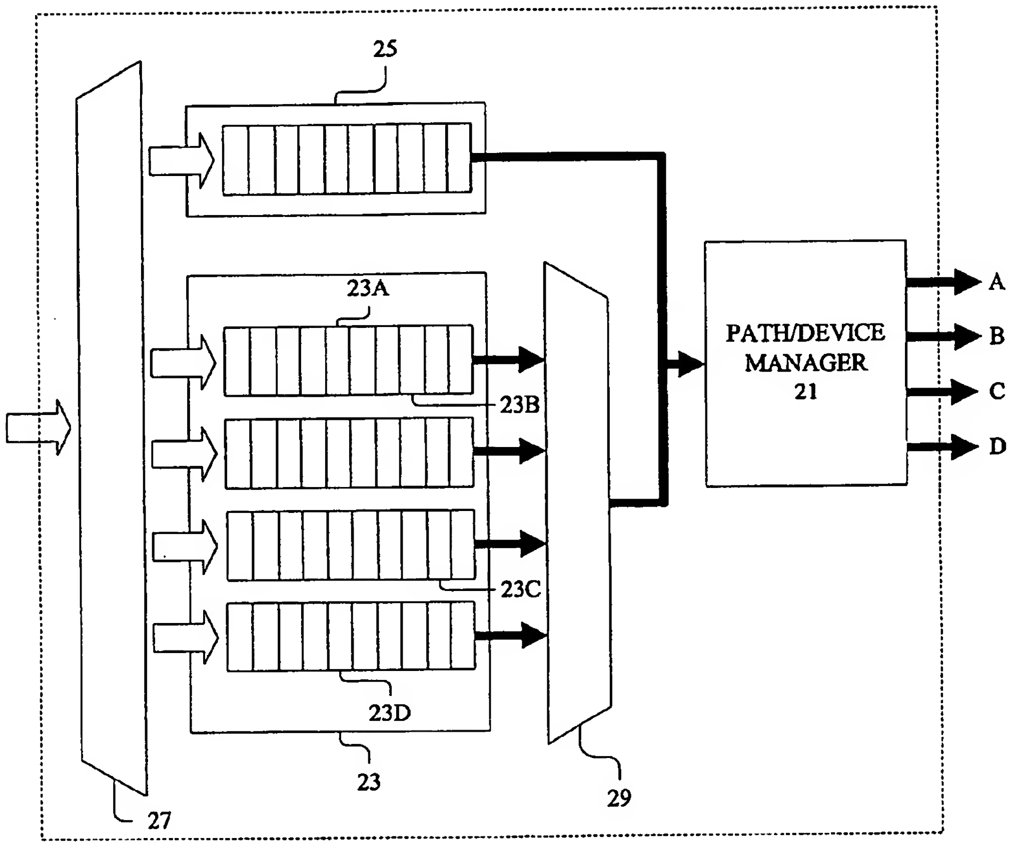
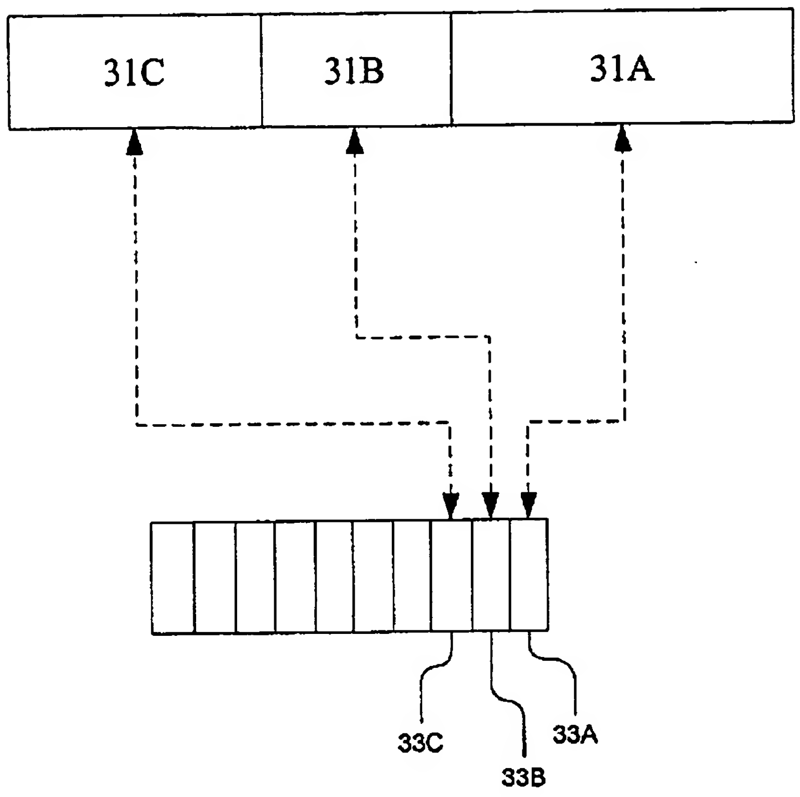


FIG. 2B



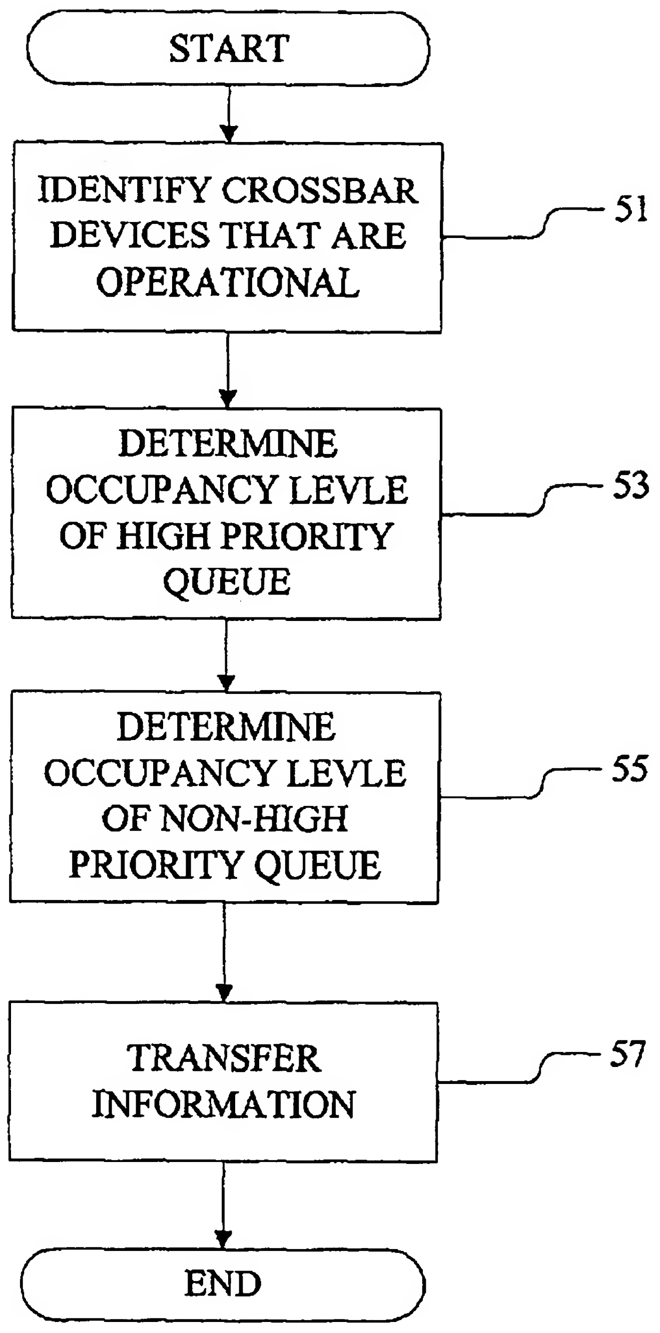
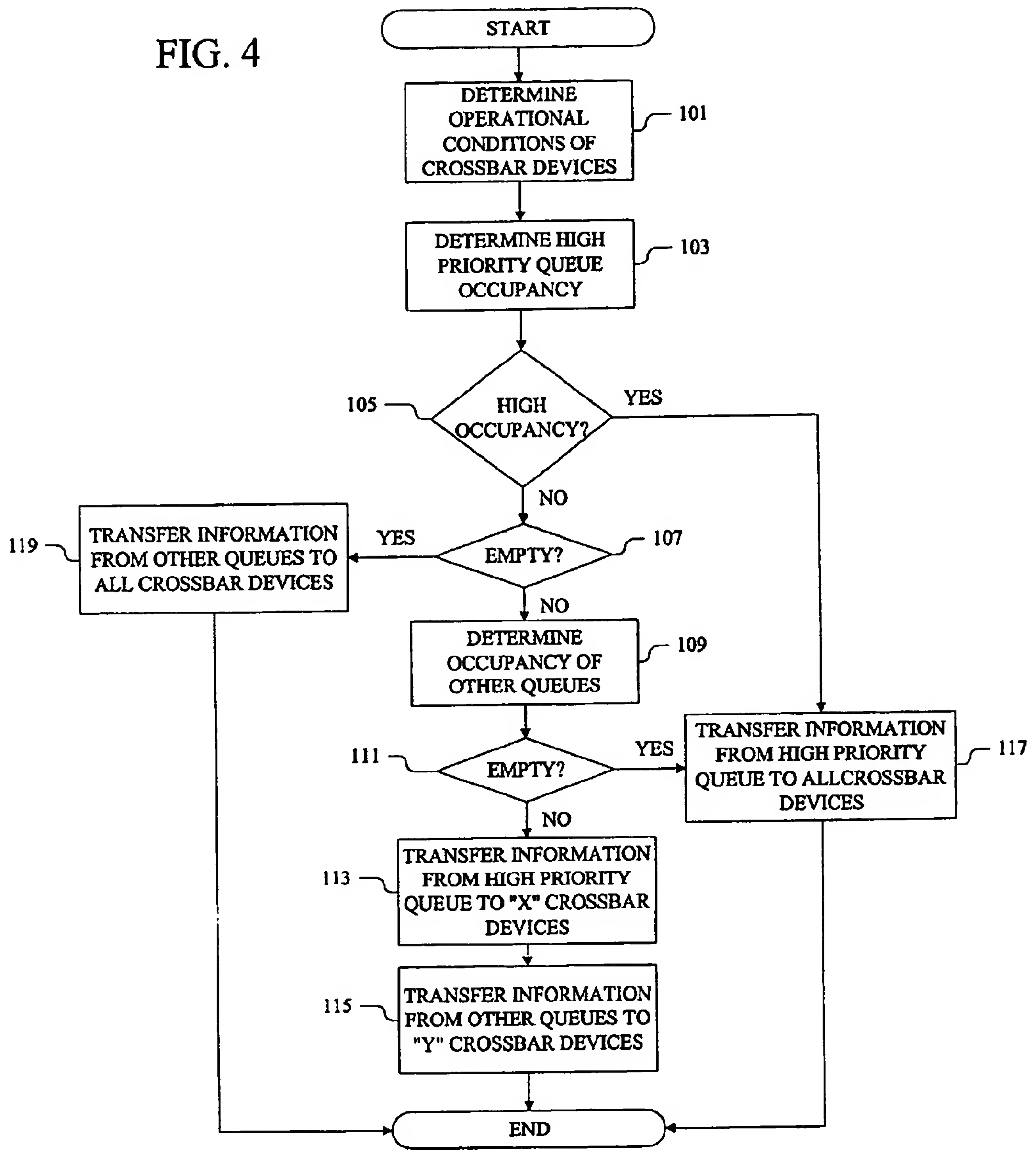

START
IDENTIFY CROSSBAR DEVICES THAT ARE OPERATIONAL
51
DETERMINE OCCUPANCY LEVLE OF HIGH PRIORITY QUEUE
53
DETERMINE OCCUPANCY LEVLE OF NON-HIGH PRIORITY QUEUE
55
TRANSFER INFORMATION
57
END

FIG. 3

FIG. 4


START
DETERMINE OPERATIONAL CONDITIONS OF CROSSBAR DEVICES
101
DETERMINE HIGH PRIORITY QUEUE OCCUPANCY
103
105
HIGH OCCUPANCY?
YES
NO
TRANSFER INFORMATION FROM OTHER QUEUES TO ALL CROSSBAR DEVICES
119
YES
EMPTY?
107
NO
DETERMINE OCCUPANCY OF OTHER QUEUES
109
111
EMPTY?
YES
TRANSFER INFORMATION FROM HIGH PRIORITY QUEUE TO ALLCROSSBAR DEVICES
117
NO
TRANSFER INFORMATION FROM HIGH PRIORITY QUEUE TO "X" CROSSBAR DEVICES
113
TRANSFER INFORMATION FROM OTHER QUEUES TO "Y" CROSSBAR DEVICES
115
END

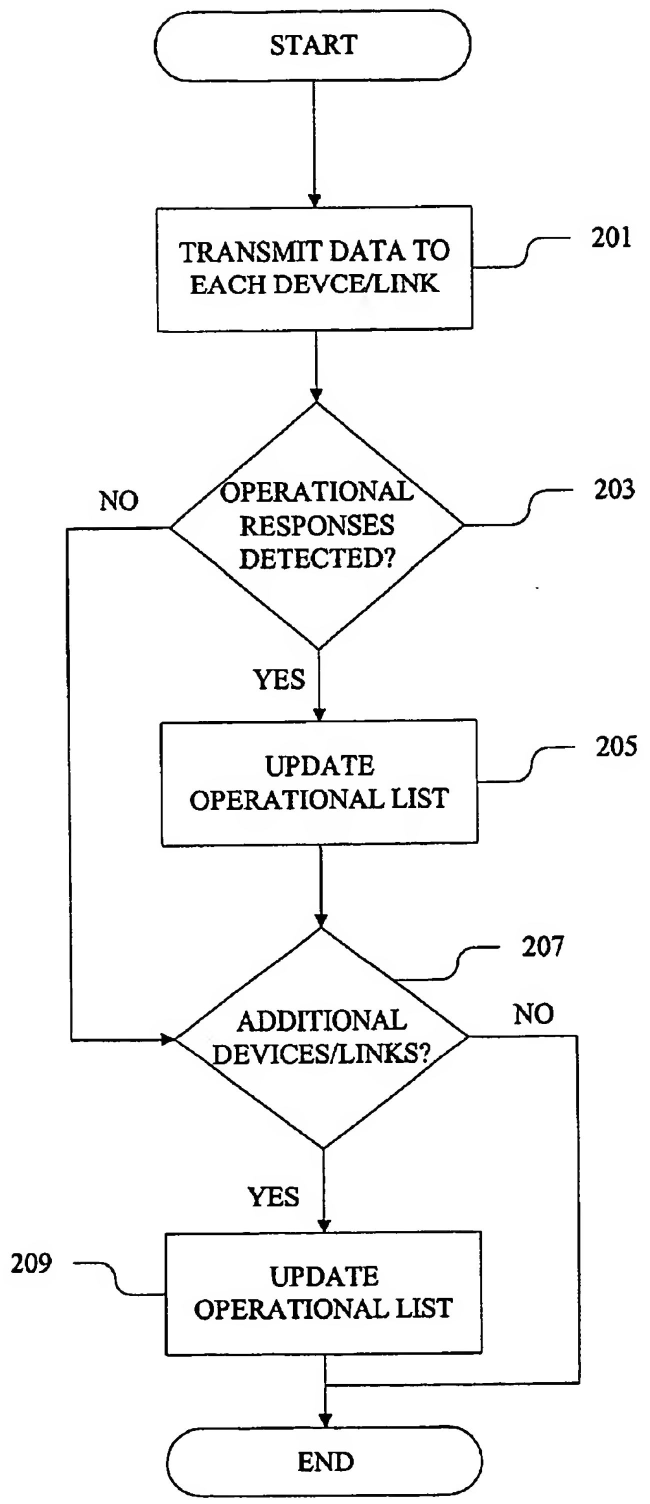

START
TRANSMIT DATA TO EACH DEVCE/LINK
201
OPERATIONAL RESPONSES DETECTED?
203
NO
YES
UPDATE OPERATIONAL LIST
205
ADDITIONAL DEVICES/LINKS?
207
NO
YES
209
UPDATE OPERATIONAL LIST
END

FIG. 5

AUTOMATIC LOAD BALANCING IN SWITCH FABRICS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Application No. 60/289,557 filed May 7, 2001 which is hereby incorporated by reference as if set forth in full herein.

BACKGROUND

[0002] The present invention relates generally to switching devices, and more particularly to load balancing a switch system with multiple switching elements, including those in which switching elements are dynamically added or removed.

[0003] Conventionally, network communication systems include multiple communication or network nodes interconnected together to provide high speed communication throughout the systems. These communication systems have become widely pervasive and are rapidly growing. However, with this growth, the demand to provide information faster without any undue delay is also growing. Likewise, the demand to provide larger amounts of information is increasing. As such, communication nodes or devices are expected to operate quicker in order to provide information faster and/or accommodate large amounts of information, i.e., support an increased bandwidth. However, at times, providing information faster and support a large bandwidth are competing demands.

[0004] Also, in order to meet these demands the ability for communication devices to be upgraded, adapted or replaced becomes a concern. Often times, communication devices are re-configured or replaced and thereby causing down-time, i.e., making a particular network inoperable for a period of time. Furthermore, the cost of upgrading and maintaining communication devices that are larger and faster may be cost prohibitive. In addition, communication devices that are under utilized become a waste of resources and in some cases may obviate the need to expand or replace a communication device to provide a particular bandwidth and/or speed.

[0005] However, in providing more information faster communication devices are also expected to maintain a particular quality of service and reliability. In other words, not only does information need to be sent and received, but the information should be sent within a specific time frame with minimal errors.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] FIG. 1 illustrates one embodiment of the load balancing system of the present invention;

[0007] FIG. 2A illustrates one embodiment of the load balancer of the present invention;

[0008] FIG. 2B illustrates one embodiment of a queue;

[0009] FIG. 3 illustrates an overview of the load balancing process of the present invention;

[0010] FIG. 4 illustrates one embodiment of the load balancing process of the present invention; and

[0011] FIG. 5 illustrates one embodiment of the determination of the operating conditions process of the present invention.

SUMMARY OF THE INVENTION

[0012] The present invention provides a load balancing method and system for a switch system with multiple switching elements or network nodes. In one embodiment, the load balancing system includes a plurality of crossbar devices and a plurality of queues. The plurality of queues are configured to receive data. The system also includes a load balancer coupled to the plurality of queues and configured to determine an amount of data in each of the plurality of queues and to send the data to specific ones of the plurality of crossbar devices based on the amount of data in each queue. In one aspect of the invention, the plurality of queues includes a high priority queue and a plurality of non-high priority queues. Also, in another aspect of the invention, the load balancer sends the data to specific crossbar devices of the plurality of crossbar devices based on the amount of data in the high priority queue relative to the amount of data in each of the plurality of non-high priority queues. Furthermore, in a further aspect of the invention, the load balancer is configured to detect inoperable crossbar devices and to detect additional crossbar devices added to the plurality of crossbar devices.

[0013] In another embodiment, a load balancing method is provided in which a plurality of data is received and stored in a plurality of queues. Each data of the plurality of data is placed in a specific queue of the plurality of queues based on a priority associated with each data. The occupancy levels in each of the plurality of queues is determined and the data is transmitted to a plurality of crossbar devices based on the determined occupancy levels in each queue.

[0014] In a further embodiment, a load balancing system is provided that includes a switching element means. The load balancing system also includes a first holding means for receiving and storing high priority data and a second holding means for receiving and storing non-high priority data. The load balancing system further includes a balancing means for determining an occupancy level of the first and second storing means and sending data to specific switching element means based on the determined occupancy level of the first storing means in relation to the determined occupancy level of the second storing means.

[0015] Many of the attendant features of this invention will be more readily appreciated as the same becomes better understood by reference to the following detailed description and considered in conjunction with the accompanying drawings.

DETAILED DESCRIPTION

[0016] Crossbar devices are specialized communication devices that provide communication paths from multiple inputs to multiple outputs. Each communication path has a predefined capacity that represents the maximum amount of information or data that can be accommodated or supported by the communication path. Crossbar devices are often found in network devices or nodes, such as switches or routers, that receive information and reconfigure a communication path in order to send the received information to a recipient designated by the received information. The present invention, generally, allows for maximum utilization of the crossbar devices and the addition of crossbar devices without user intervention and down-time.

[0017] FIG. 1 shows one embodiment of a switch configuration of the present invention. Multiple data streams are routed through a node. The node includes network and/or packet processors **11** and, in some embodiments, various other specialized processors or circuitry for performing various processes associated with transmission, reception, and routing of, for example, ATM or SONET traffic. The processors process data incoming to and outgoing from the node. In one embodiment, the processors perform network traffic management functions. Within the node, the processors provide data to and receive data from transceivers, with the data being provided to the transceivers **13**, for example, over a common switch interface (CSIX)-L1 compliant link.

[0018] The transceivers are coupled to multiple crossbar devices **15** in parallel. In one embodiment, each of the transceivers include a plurality, such as four, i/o ports available for coupling to the crossbars. The crossbars route data from one transceiver to another. The use of multiple crossbar devices allows for increased data throughput, as well as for redundancy in the event of failure of a crossbar device.

[0019] In operation, the switch in one embodiment is under centralized control of a processor. The processor receives routing requests and priority requests from transceivers, allocates crossbars, and provides configuration commands to the crossbars. In other embodiments, the transceivers perform distributed control of the switch.

[0020] The crossbars and the transceivers may have different clock domains. In such a configuration various alignment methods may be used, for example such as those described in U.S. Pat. Application HIGH SPEED CROSS POINT SWITCH ROUTING CIRCUIT WITH WORD-SYNCHRONOUS SERIAL BACK PLANE, application Ser. No. 09/129,662; Filed Aug. 5, 1998, the disclosure of which is incorporated in its entirety herein.

[0021] The switch includes an automatic load balancing function. In one embodiment, a load balancer **17** performs the load balancing function. This load balancer can be used with 1 to N crossbar switch fabric devices. The load balancer determines which crossbar devices are active and evenly distributes the data traffic across these devices. This allows easy fabric bandwidth upgrades by simply adding more crossbar devices. In addition, redundant crossbar devices that are normally left unused, can add extra fabric bandwidth during normal operation.

[0022] The load balancer is also configured to automatically stop sending data to a failed crossbar device. In one embodiment the transceivers determine whether a crossbar device is operational by monitoring responses to an initialization command sent to the crossbar via one of the i/o ports. If the crossbar responds to the initialization command and is able to perform alignment functions, if necessary, the crossbar is available for use. If the crossbar is available for use, the crossbar is allocated for data transmission by the load balancer. If the crossbar is not available for use, the crossbar is not allocated for data transmission by the load balancer.

[0023] The transceiver is able to determine when a crossbar is no longer available through a variety of mechanisms, such as periodic status messages provided by the crossbar. In one embodiment, however, the crossbars receive as part of the data transmissions periodic word or frame alignment information. The crossbars perform, for example, frame alignment using the alignment information, and generate an out of frame signal (OOF) (or more often a not OOF signal). When the transceivers receive an indication that a crossbar is not in alignment, the crossbar is treated as unavailable by the load balancer. In other embodiments, framing, for example, is performed by receiving transcievers, and the receiving transceivers provide indications of lack of alignment, as well as an indication of which crossbar is providing the out of alignment data.

[0024] Thus, to upgrade the fabric bandwidth by adding more crossbar devices, the system need not be reconfigured through software control. Multiple crossbar devices are also used for redundancy. Moreover, redundant crossbars may be used during normal system operation as load balancing results in the use of the remaining active crossbars.

[0025] In one embodiment, the load balancing function works as follows. Ingress queues shown in **FIG. 2A** can have from 0 to N frames ready for transmission to multiple crossbar devices, in this case four, coupled to links A, B, C or D. In the example of **FIG. 2A**, the ingress queues receives frames from a de-multiplexer **27** which receives a data stream. The queues and the links A, B, C, and D are associated with a path/device manager **29**. As illustrated, the ingress queues include a high priority (HP) queue **25** and four non-HP queues **23***a-d*. In one embodiment, the frames from the non-high priority queues are provided to the path/device manager via a multiplexer **29**. A low (L), medium (M) or high (H) threshold mark or capacity indicator is established for each queue. The occupancy level or the amount of information, e.g., the number of frames, in each queue determines the order in which data is to be transmitted across the four crossbar devices as shown, for example, in Table 1.

TABLE 1

High Priority Queue's Occupancy Level	Non-High Priority Queues' Occupancy Level	Links for the High Priority Queue	Links for the Non-High Priority Queue
High	Any*	A, B, C, D	—
Medium	High	A	B, C, D
Medium	Medium	A, B	C, D
Medium	Low	A, B, C,	D
Medium	Empty	A, B, C, D	—
Low	High	—	A, B, C, D
Low	Medium	A	B, C, D
Low	Low	A, B	C, D
Low	Empty	A, B, C, D	—
Empty	Any*	—	A, B, C, D

*"Any" being high, medium, low or empty

[0026] Examination of a simple case when only one queue has data is instructive. If the queue holds four or more frames, crossbar connections are requested from all four crossbar devices. If the queue only holds three frames, connections are requested from only three crossbar devices etc. If there are less then 4 frames in the queue, the crossbar devices are selected using a round robin method.

[0027] In most cases, more then one queue will hold data frames. Queues, in various embodiments, have different priorities and different threshold marks allowing the user to tailor the bandwidth from each queue as shown in the table above.

[0028] If one of the crossbar devices has failed or is pulled from the system, the load balancer can detect this out of frame condition through the links labeled A, B, C and D. If one of these devices coupled to these links is out of frame, the load balancer will no longer send connection requests or data to this crossbar device. This mechanism is automatic and requires no user or software intervention.

[0029] Thus, the switch fabric can be easily upgraded by adding more crossbar devices in parallel, avoiding system downtime. In addition, the switch fabric does not need to identify or dedicate crossbar devices for redundancy. All crossbar devices in the fabric can be actively used.

[0030] **FIG. 2B** illustrates one embodiment of a queue of the present invention. The queue is divided into portions, in this case, a first portion **31***a*, a second portion **31***b* and a third portion **31***c*. When data is first received it is placed in the first portion of the queue. Once the first portion of the queue is completely filled the received data is placed in the second portion of the queue. Likewise, when the second portion of the queue is completely filled with data, the received data is placed in the third portion of the queue. In other words, data will be placed in a succeeding portion of the queue once the previous portion of the queue is filled. In one embodiment, corresponding portion indicators **33***a, b, c,* are associated with the first portion, the second portion, and the third portion, respectively. The portion indicators indicate that data is in one of the three portions.

[0031] In one embodiment, the portion indicators are bits in a portion register. In this embodiment, if, for example, the second portion indicator **33***b* is high or set to a logic level one, this indicates that data is in the second portion of the queue. The portion register, in one embodiment, includes numerous portion indicators that corresponds to numerous portions for each of the queues. By using the portion indicators for determining the occupancy level or the amount of data or information in each queue, the order in which data is transmitted to the cross-bar devices is determined. Similar to the determination shown in Table 1, the information may be transmitted from a queue to a particular device via a link by using the first portion indicator for the low capacity indicator, the second portion indicator for the medium capacity indicator and the high capacity indicator for the third portion indicator.

[0032] In various embodiments, the queue is also divided into more than three portions or less than three portions and/or has corresponding portion indicators associated with each portion of the queue. Also, the size of each portion, e.g., the amount of frames capable of being in each portion, may vary and may be modified and/or configured by a user or the load balancer. Furthermore, the boundaries of the portions are set, determined and/or modified by a user or the load balancer, for example, by adjusting the association of the portion indicators to the corresponding portion.

[0033] In **FIG. 3**, an overview of the load balancing function is illustrated. In block **51**, the process determines which crossbar devices are operational. In block **53**, the process determines the occupancy level of the high priority queue. In block **55**, the process determines the occupancy level of the non-high priority queue. In block **57**, the process arranges for the transfer of information from the high priority queues and the non-high priority queues to the operational crossbar devices based on the occupancy levels of the queues and then the process ends.

[0034] In **FIG. 4**, one embodiment of the load balancing function is illustrated. In block **101**, the process determines the operational conditions of the crossbar devices. The number of crossbar devices that are operational is recorded. For example, the total number of crossbar devices may be five and the number of operational crossbar devices may be three, e.g., crossbar devices 1, 2 and 3. In block **103**, the process determines the occupancy level of the high priority queue. In one embodiment, the process reads an indicator register of the high priority queue to determine the occupancy level of the queue, e.g., high, medium, low or empty. In block **105**, the process determines if the occupancy level of the high priority queue is high. If the process determines that the occupancy level of the high priority queue is high, the process in block **117** assigns or arranges that the information in the high priority queue is transferred to all the operational crossbar devices determined in block **101**.

[0035] If the process determines that the occupancy level of the high priority queue is not high, the process in block **107**, determines if the occupancy level of the high priority queue is empty. In one embodiment, the process determines that the occupancy level is not high, medium or low to determine that the queue is empty. If the process determines that the high priority queue is empty, the process arranges that the information in the non-high priority queues are transferred to all the operational crossbar devices in block **119** and the process ends.

[0036] If the process determines that the high priority queue is empty, the process in block **109** determines the occupancy level of the other non-high priority queues. In one embodiment, the process accesses/reads an indicator register of the high priority queue to determine the occupancy level of the queue, e.g., high, medium, low or empty. In block **111**, the process determines if the other queues are empty. If the process determines that the other queues are empty, the process arranges to transfer information from the high priority queue to all the operational crossbar devices, in block **117**, and the process ends. In one embodiment, the process does not end, but continues, repeating over again at block **101** until commanded externally to end, such as by a shutdown command provided by a user or a control hardware/software.

[0037] If the process, in block **111**, determines that the other queues are not empty, the process in block **113** arranges to transfer information from the high priority queue to some of the operational crossbar devices, e.g., crossbar devices A and B. The process in block **115** arranges to transfer information from the other non-high priority queues to the other remaining operational crossbar devices, e.g., crossbar device C, and then the process ends. In one embodiment, the process in blocks **113** and **115** determines the arrangement of the transfer of information from the high priority and non-high priority queues to the operational crossbar devices based on the following:

[0038] When the high priority queue has an occupancy level of medium or low, for each occupancy level (from high to low) of the non-high priority queues the process allocates one less crossbar devices. For example, if the occupancy level is high for the non-high priority queues, the process allocates four crossbar devices, for a medium occupancy level, three crossbar devices, and for a low occupancy level, two crossbar devices. The occupancy level of the high priority queue determines the starting point or total number of crossbar devices that the process allocates to the non-high priority queues. Using the above example, if the occupancy level of the high priority queue is low, the total number of crossbar devices is four and for a medium occupancy level, the total number of crossbar devices is three. If the total number of operational crossbar devices is four, then in the above example, the information in the high priority queue is not transferred to a crossbar device until the occupancy level of the non-high priority queues reduces from high to medium.

[0039] In one embodiment, redundancy is employed by utilizing an additional spare or a complete set of spare links or crossbar devices (e.g., for each available link or crossbar device another link or crossbar device acting as a backup is provided) dedicated to handle the transfer of information if a link or crossbar device fails. As such, the switch-over from one link or crossbar device that failed to the additional (redundant) link or crossbar device is performed quickly to cause minimal or no down-time by the switch. In another embodiment, redundancy is employed utilizing all the available links or crossbar devices that are operational. If a link or crossbar device fails, information transfer continues using the number of available links or crossbar devices, however, the number of available links or crossbar devices is now reduced by one, i.e., minus the failed link or device. Thus, the transfer of information is maintained and thereby causing minimal or no down-time to the switch.

[0040] Similarly, in one embodiment, scaleability is employed by again utilizing all the available links or crossbar devices that are operational. If a link or crossbar device is added, information transfer continues using the number of available links or crossbar devices, however, the number of available links or crossbar devices is now increased by one or more, i.e., adding the new links or devices. Hence, transfer of information is maintained and thus causing minimal or no down-time to the switch.

[0041] FIG. 5 illustrates an embodiment of the process of determining the operating conditions of the available crossbar devices or links. In block **201**, the process transmits predetermined data, frames or packets, to each crossbar device or link. In one embodiment, the predetermined data comprise a status command, framing packets and/or information to be transmitted on the device or link. In one embodiment, the load balancer sends the data to the devices or links. In block **203**, the process detects any operational responses from the crossbar devices or links in response to the predetermined data. In one embodiment, the process waits for a predetermined response such as an out of frame error, a status response or an error message, to indicate that a particular device or link or a set of devices or links are not operating or available to receive information. In another embodiment, the process waits for a specific period of time and if no response is received the process determines that an operational response has occurred.

[0042] If, in block **203**, the process detects an operational response, the process updates an operational list of crossbar devices in block **205**. In one embodiment, the operational list comprises a list of available crossbar devices or available links and the process updates the operational list by removing the crossbar device or link from the list when an operational response, e.g., an out of frame condition, is detected in block **203**. The operational list, in one embodiment, is utilized by the load balancer or process both previously described to determine the operational conditions of the devices or links for the transmission of information. In one embodiment, the process updates the list by disabling a device or link when a non-operational link or device is detected by, for example, marking or otherwise indicating in the list that the device is not operational. Also, in one embodiment, the mark or indication is recognized by the load balancer or the process to determine the operational conditions of the devices or links. If the process does not detect an operational response, the process continues to block **207**.

[0043] In block **207**, the process determines if additional devices or links are available. In one embodiment, the process determines if additional devices or links are available by transmitting predetermined data to a location where an additional device or link might be. In various embodiments, the location where an additional device or link is expected to be is a predefined offset from the last operational crossbar device or link detected, a specific address location or shared memory space where all the devices and links are provided a specific address/memory space or location, or an additional entry in the operational list. If, in block **207**, the process does not detect an operational response in response to the transmitted data, the process updates the operational list in block **209**. The process then ends. In one embodiment, the process adds the additional link or device located to the operational list. If, in block **207**, the process does detect an operational condition indicating that a new or additional device or link is not operating, e.g., provides an out of frame error, then the process ends.

[0044] In one embodiment, operations performed by the process in blocks **207-209** are combined with blocks **201-205**. For instance, the process in block **201** transmits predetermined data to each device or link and additionally to a location where an additional device or link might be. In block **203**, the process detects an operational response, e.g., an out of frame condition, in response to the transmitted data from each of the devices or links and any possible additional new devices or links. In block **205**, the process updates the operational list, e.g., removing or disabling devices or links from the list that are not operational, e.g., having an out of frame condition, and/or adding devices or links to the list that are new and operational. In one embodiment, the process continually repeats at predetermined intervals, such as once a day or once a week, or at specific predetermined times, such as at initialization or a scheduled maintenance. In another embodiment, the process does not send or transmit predetermined packets, but waits to be interrupted or polls for a response from the devices or links as to the operational condition of each device or link.

[0045] In one embodiment, the operational response indicates that a particular device or link or a set of devices or links are operational or available to receive information. Thus, in this embodiment, in blocks **205** and **209**, if the process detects an operational condition, the process, in one embodiment, updates the operational list by adding the device or link that provided the response or the device or link indicated as being operational by the response. Likewise, if the process does not detect an operational condition, the process removes or disables the device or link in the operational list. Also, in one embodiment, the process in block **205** may be configured to detect an operational condition that indicates that a particular device or link or set of devices or links are operational or available to receive information and in block **209**, the process may be configured to detect an operational condition that indicates that a particular device or link or set of devices or links are not operational or not available to receive information or vice versa.

[0046] Accordingly, the present invention provides an automatic load balancer. Although the invention has been described in certain specific embodiments, it should be recognized that those of skill in the art would recognize other insubstantially different ways to implement the present invention. Thus, the present embodiments should be considered exemplary only, with the invention defined by the claims and their equivalents supported by this disclosure.

What is claimed is:

1. A load balancing system for network nodes, the load balancing system comprising:

a plurality of crossbar devices;

a plurality of queues configured to receive data; and

a load balancer coupled to the plurality of queues and configured to determine an amount of data in each of the plurality of queues and to send the data to specific ones of the plurality of crossbar devices based on the amount of data in each queue.

2. The load balancing system of claim 1 wherein the plurality of queues comprises a high priority queue and a plurality of non-high priority queues.

3. The load balancing system of claim 2 wherein the load balancer sends data to specific crossbar devices of the plurality of crossbar devices based on an amount of data in the high priority queue relative to an amount of data in each of the plurality of non-high priority queues.

4. The load balancing system of claim 2 wherein the load balancer sends data to specific crossbar devices in a order that is based on one of the amount of data in the high priority queue and an amount of data in each of the plurality of non-high priority queues.

5. The load balancing system of claim 2 wherein the load balancer sends data to specific crossbar devices of the plurality of crossbar devices based on an amount of data in each of the plurality of non-high priority queues relative to an amount of data in the high priority queue.

6. The load balancing system of claim 2 wherein the load balancer sends data to specific crossbar devices in a order based on one of an amount of data in the high priority queue and an amount of data in each of the plurality of non-high priority queues.

7. The load balancing system of claim 1 further comprising a capacity indicator identifying the amount of data in each queue.

8. The load balancing system of claim 7 wherein the load balancer is configured to determine the amount data in each queue based on examining the capacity indicator and to transmit data to the plurality of crossbar devices in a predetermined order based on the examination of the capacity indicator.

9. The load balancing system of claim 7 wherein the capacity indicator further indicates an occupancy level based on the amount of data in each queue.

10. The load balancing system of claim 9 wherein load balancer is configured to transmit data to the plurality of crossbar devices in a predetermined order based on various occupancy levels of each queue as indicated by the capacity indicator.

11. The load balancing system of claim 10 wherein the occupancy levels are high, medium, low and empty.

12. The load balancing system of claim 1 wherein each of the queues are divided into a plurality of portions having a corresponding portion indicator for each portion of the queues to identify that data are in a corresponding portion of a queue.

13. The load balancing system of claim 12 wherein the portion indicators are modifiable to indicate various occupancy levels in the queue.

14. The load balancing system of claim 12 wherein the load balancer is configured to transmit data from the plurality of queues to the plurality of crossbar devices in a predetermined order based on the portion indicators.

15. The load balancing system of claim 2 wherein each of the queues are divided into a first portion, a second portion and a third portion.

16. The load balancing system of claim 15 wherein the data received are placed first in the first portion, when the first portion is full, the received data are placed in the second portion and, when the second portion is full, the received data are placed the third portion of the queue.

17. The load balancing system of claim 15 wherein the load balancer, upon determining that data are in the third portion of the queue of the high priority queue, causes the data in the high priority queue to be transmitted to all the plurality of crossbar devices that are available.

18. The load balancing system of claim 15 wherein the load balancer, upon determining that data are in the first portion of the queue of the high priority queue, causes the data in the third portion of the non-high priority queues to be transmitted to all the plurality of crossbar devices that are available.

19. The load balancing system of claim 15 wherein the load balancer, upon determining that data are only in the non-high priority queues, causes the data in the non-high priority queues to be transmitted to all the plurality of crossbar devices that are available.

20. The load balancing system of claim 15 wherein the load balancer, upon determining that data are only in the high priority queues, causes the data in the high priority queues to be transmitted to all the plurality of crossbar devices that are available.

21. The load balancing system of claim 15 wherein the load balancer, upon determining that data are in one of the first and second portions of the queue of the high priority queue, causes the data in the non-high priority queues to be transmitted to particular predetermined crossbar devices that are available and causes the data in the high priority queue to be transmitted to remaining crossbar devices from the plurality of crossbar devices that are available.

22. The load balancing system of claim 15 further comprising:

a first indicator identifying that data are in the first portion of a queue;

a second indicator identifying that data are in the second portion of a queue; and

a third indicator identifies that data are in the third portion of a queue.

23. The load balancing system of claim 15 wherein the load balancer, upon determining that the third indicator identifies that data are in the third portion of the queue of the high priority queue, causes the data in the non-high priority queues to be transmitted to all the plurality of crossbar devices that are available.

24. The load balancing system of claim 1 wherein the load balancer is configured to detect inoperable crossbar devices.

25. The load balancing system of claim 24 wherein the load balancer in detecting inoperable devices comprises sending a message to the plurality of crossbar devices and receiving a response sent from each of the plurality of crossbar devices that are operating.

26. The load balancing system of claim 24 wherein the load balancer in detecting inoperable devices comprises sending a message to the plurality of crossbar devices and determining if a response sent from each of the plurality of crossbar devices that are operating based on a predetermined time frame.

27. The load balancing system of claim 24 wherein the load balancer is configured to detect additional crossbar devices added to the plurality of crossbar devices.

28. The load balancing system of claim 27 wherein the load balancer in detecting additional crossbar devices comprises sending a data to at least one predetermined location and receiving a response sent from each of the additional crossbar devices that are added.

29. The load balancing system of claim 27 wherein the load balancer in detecting additional crossbar devices comprises receiving a data sent from each of the additional crossbar devices that are added.

30. The load balancing system of claim 25 wherein the predetermined location is an offset from one of the plurality of crossbar devices.

31. The load balancing system of claim 1 further comprising a processor coupled to the load balancer.

32. A load balancing method comprising:

receiving a plurality of data;

storing the plurality of data in a plurality of queues, each data of the plurality of data being placed in a specific queue of the plurality of queues based on a priority associated with each data;

determining occupancy levels in each of the plurality of queues; and

transmitting the data to a plurality of crossbar devices based on the determined occupancy levels in each queue.

33. The load balancing method of claim 32 wherein the plurality of queues comprises a high priority queue and a plurality of non-high priority queues.

34. The load balancing method of claim 33 wherein the occupancy levels in each queue are based on an amount of data in the high priority queue and an amount of data in each of the plurality of non-high priority queues.

35. The load balancing method of claim 33 wherein transmitting the data, the data are transmitted to specific crossbar devices in an order that is based on an occupancy level of the high priority queue and an occupancy level in each of the plurality of non-high priority queues.

36. The load balancing method of claim 32 further comprising dividing the queues into a plurality of portions having a corresponding portion indicator for each portion of the queues to identify that data are in a corresponding portion of a queue.

37. The load balancing method of claim 32 further comprising modifying the portion indicators to indicate various occupancy levels in the queue.

38. The load balancing method of claim 32 wherein transmitting the data, the data are transmitted to specific crossbar devices in an order based on the portion indicators.

39. The load balancing method of claim 33 further comprising:

dividing each of the queues into a first portion, a second portion and a third portion; and

determining if data are in the first, second and third portions of the plurality of queues.

40. The load balancing method of claim 39 wherein the transmitting of the data further comprises transmitting the data in the high priority queue to all the plurality of crossbar devices, upon determining that data are in the third portion of the queue of the high priority queue.

41. The load balancing method of claim 39 wherein the transmitting of the data further comprises transmitting the data in the third portion of the non-high priority queues to all the plurality of crossbar devices, upon determining that data are in the first portion of the queue of the high priority queue.

42. The load balancing method of claim 39 wherein the transmitting of the data further comprises transmitting the data in the non-high priority queues to all the plurality of crossbar devices, upon determining that data are only in the non-high priority queues.

43. The load balancing method of claim 39 wherein the transmitting of the data further comprises transmitting the data in the high priority queues to all the plurality of crossbar devices, upon determining that data are only in the high priority queues.

44. The load balancing method of claim 39 wherein the transmitting of the data further comprises transmitting the data in the non-high priority queues to particular predetermined crossbar devices and transmitting the data in the high priority queue to remaining crossbar devices from the plurality of crossbar devices, upon determining that data are in one of the first and second portions of the queue of the high priority queue.

45. The load balancing method of claim 32 further comprising detecting one of inoperable crossbar devices and operable crossbar devices and transmitting data to the crossbar devices based on the detecting of one of inoperable crossbar devices and operable crossbar devices.

46. The load balancing method of claim 32 further comprising detecting an operational condition of each of the plurality of crossbar devices and transmitting data to the crossbar devices based on the operational condition detected.

47. The load balancing method of claim 32 further comprising detecting additional crossbar devices added to the plurality of crossbar devices and transmitting the data to the detected additional crossbar devices.

48. A load balancing system comprising:

switching element means;

first holding means for receiving and storing high priority data;

second holding means for receiving and storing non-high priority data; and

balancing means for determining an occupancy level of the first and second storing means and sending data to specific switching element means based on the determined occupancy level of the first storing means in relation to the determined occupancy level of the second storing means.

* * * * *

US006249528B1

(12) United States Patent
Kothary

(10) Patent No.: US 6,249,528 B1
(45) Date of Patent: Jun. 19, 2001

(54) NETWORK SWITCH PROVIDING PER VIRTUAL CHANNEL QUEUING FOR SEGMENTATION AND REASSEMBLY

(75) Inventor: **Piyush Kothary**, San Jose, CA (US)

(73) Assignee: **I-Cube, Inc.**, Campbell, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/041,826**

(22) Filed: **Mar. 12, 1998**

(51) **Int. Cl.**^{7} **H04J 3/16**
(52) **U.S. Cl.** **370/466**; 370/395; 370/474
(58) **Field of Search** 370/395, 465, 370/466, 412, 428, 474

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,805,805	*	9/1998	Civanlar et al.	370/409
6,052,383	*	4/2000	Stoner et al.	370/466

* cited by examiner

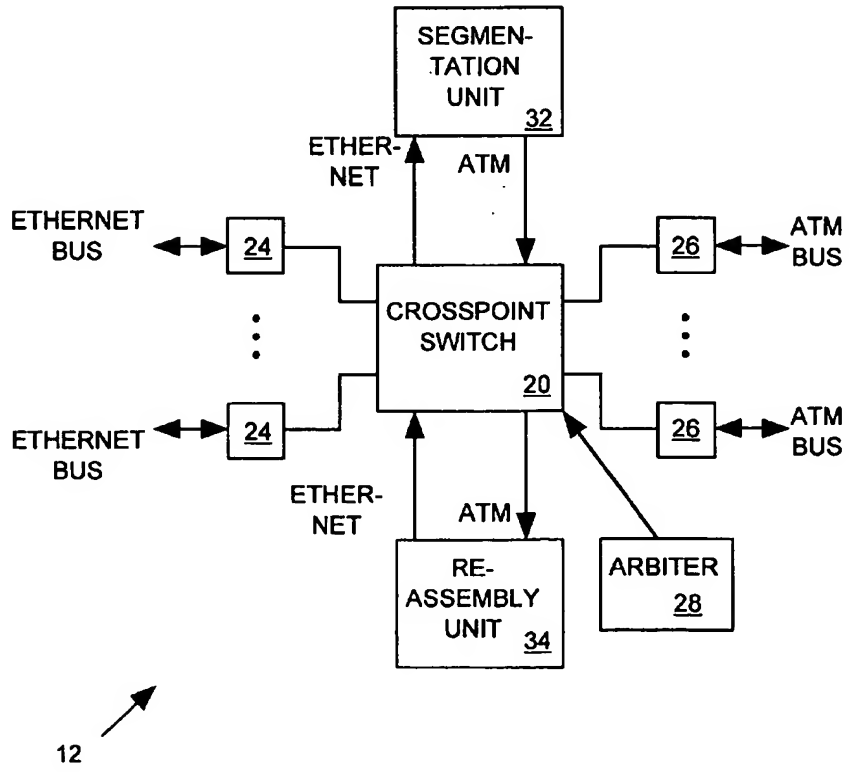
Primary Examiner—Wellington Chin
Assistant Examiner—Maikhanh Tran

(74) *Attorney, Agent, or Firm*—Daniel J. Bedell; Smith-Hill and Bedell

(57) **ABSTRACT**

A network routing switch includes a crosspoint switch, a set of Ethernet I/O ports, a set of ATM I/O ports, a reassembly unit for converting ATM transmissions into Ethernet transmissions, and a segmentation unit for converting Ethernet transmissions into ATM transmissions. As an ATM I/O port receives cells of an ATM transmission from an external source it stores them until the transmission is complete and then sends the ATM transmission through the crosspoint switch either to a forwarding ATM port or to the reassembly unit depending on whether the ATM transmission is to be forwarded as an ATM or Ethernet transmission. When the reassembly unit receives an ATM transmission it converts it to an Ethernet transmission and forwards it through the crosspoint switch to a forwarding Ethernet I/O port. When an Ethernet I/O port receives an Ethernet packet from an external source, it sends the packet through the crosspoint switch either to a forwarding Ethernet port or to the segmentation unit, depending on whether the Ethernet packet is to be forwarded as an Ethernet or ATM transmission. When the segmentation unit receives an Ethernet packet it converts it to an ATM transmission and forwards it through the crosspoint switch to a forwarding ATM I/O port.

21 Claims, 9 Drawing Sheets



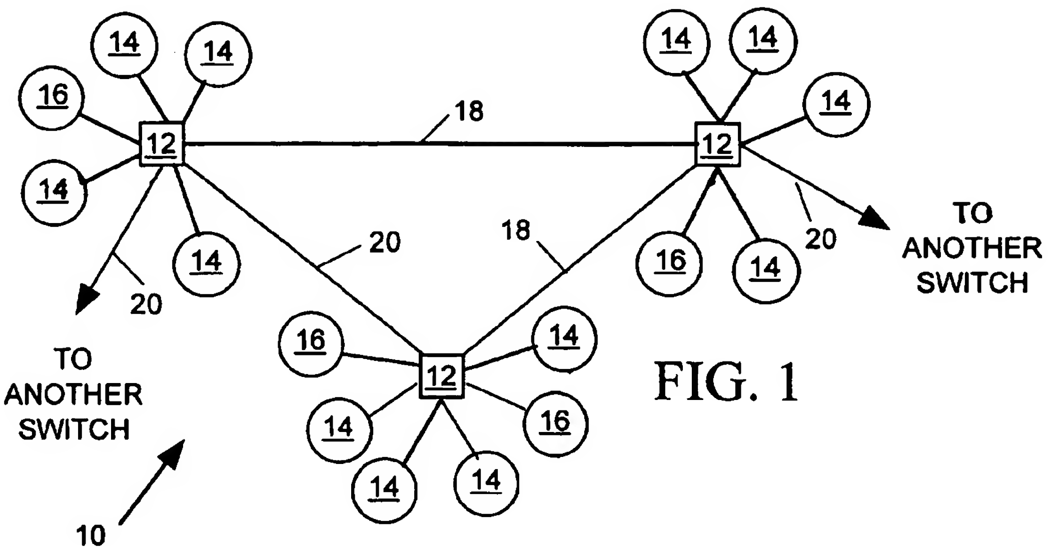

14
14
16
12
14
14
18
20
20
TO ANOTHER SWITCH
14
14
14
12
16
14
20
TO ANOTHER SWITCH
18
16
12
14
14
16
14
14
10

FIG. 1

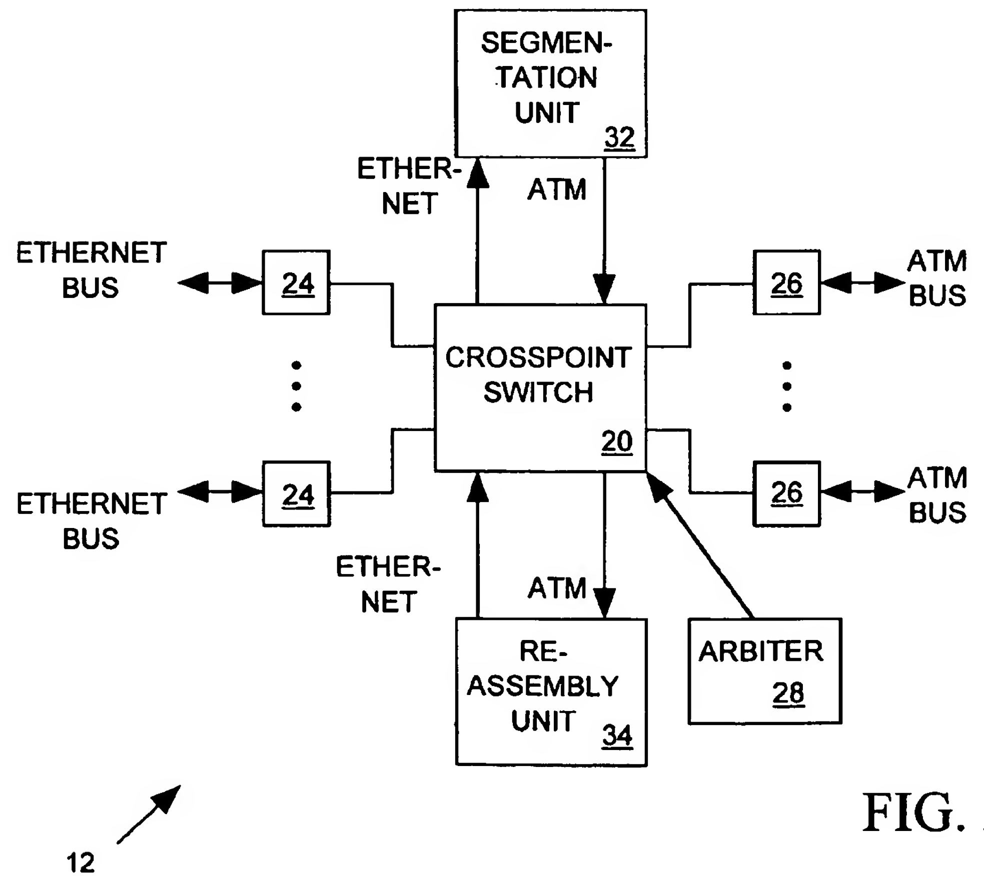

SEGMENTATION UNIT 32
ETHERNET
ATM
ETHERNET BUS
24
CROSSPOINT SWITCH 20
26
ATM BUS
ETHERNET BUS
24
26
ATM BUS
ETHERNET
ATM
RE-ASSEMBLY UNIT 34
ARBITER 28
12

FIG. 2

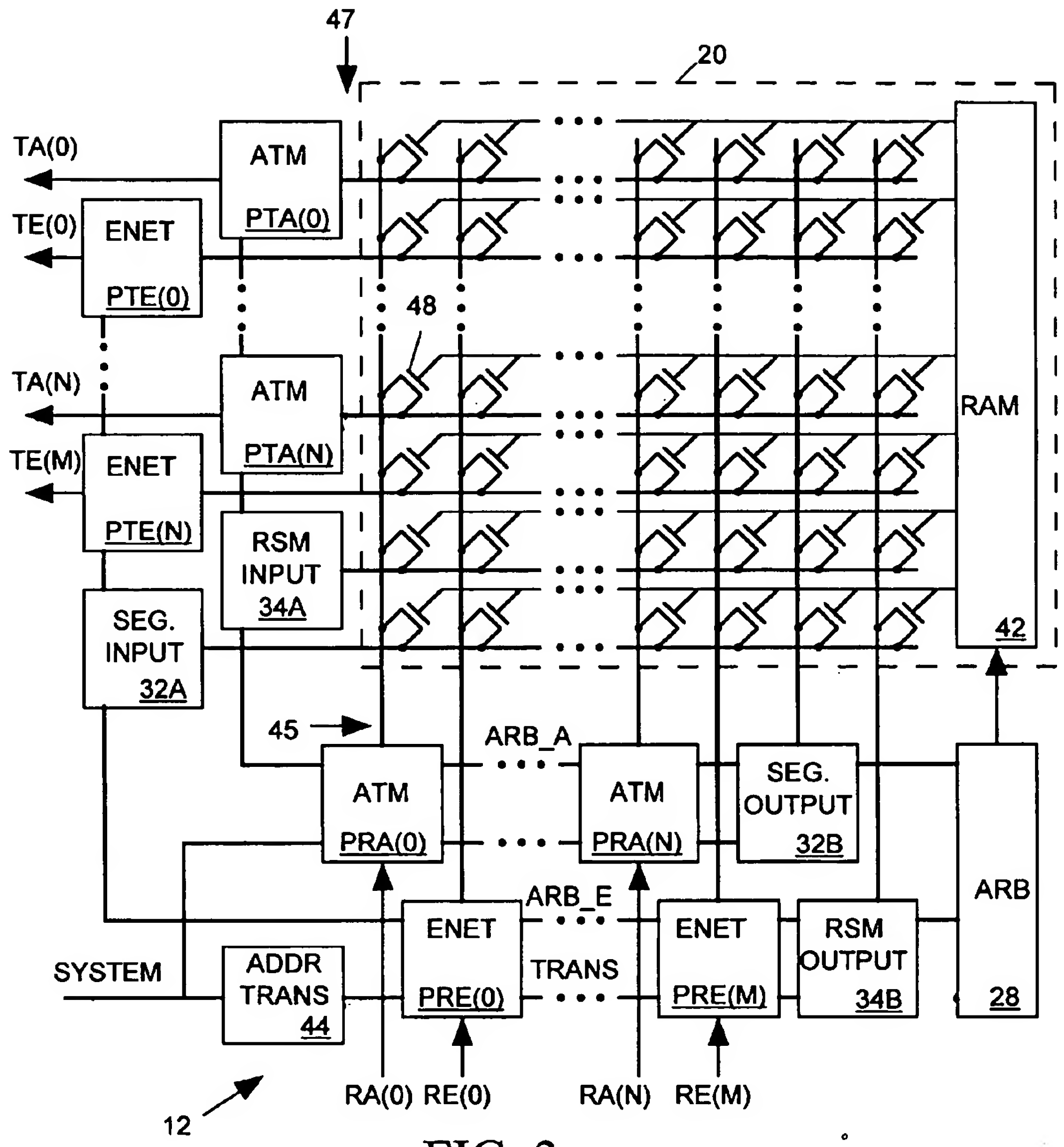


FIG. 3

FIG. 4A

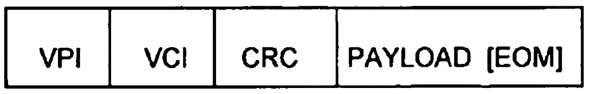
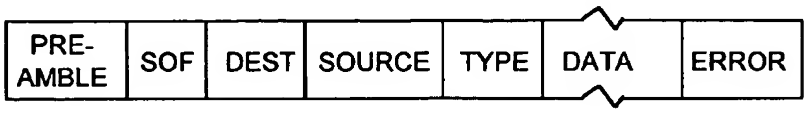


FIG. 4B



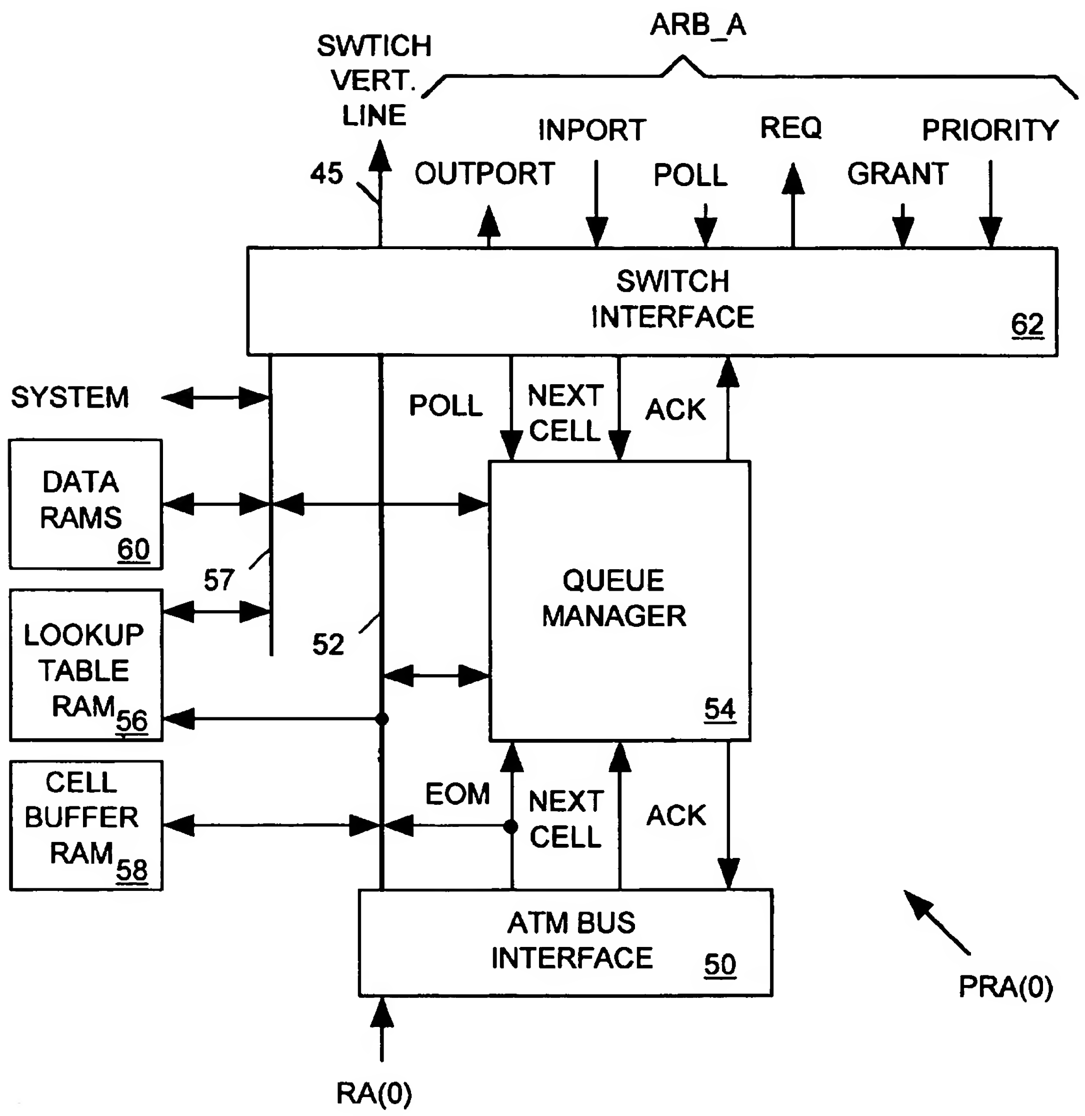

ARB_A
SWTICH VERT. LINE
45
OUTPORT
INPORT
POLL
REQ
GRANT
PRIORITY
SWITCH INTERFACE
62
SYSTEM
POLL
NEXT CELL
ACK
DATA RAMS
60
57
LOOKUP TABLE RAM
56
52
QUEUE MANAGER
54
CELL BUFFER RAM
58
EOM
NEXT CELL
ACK
ATM BUS INTERFACE
50
PRA(0)
RA(0)

FIG. 5

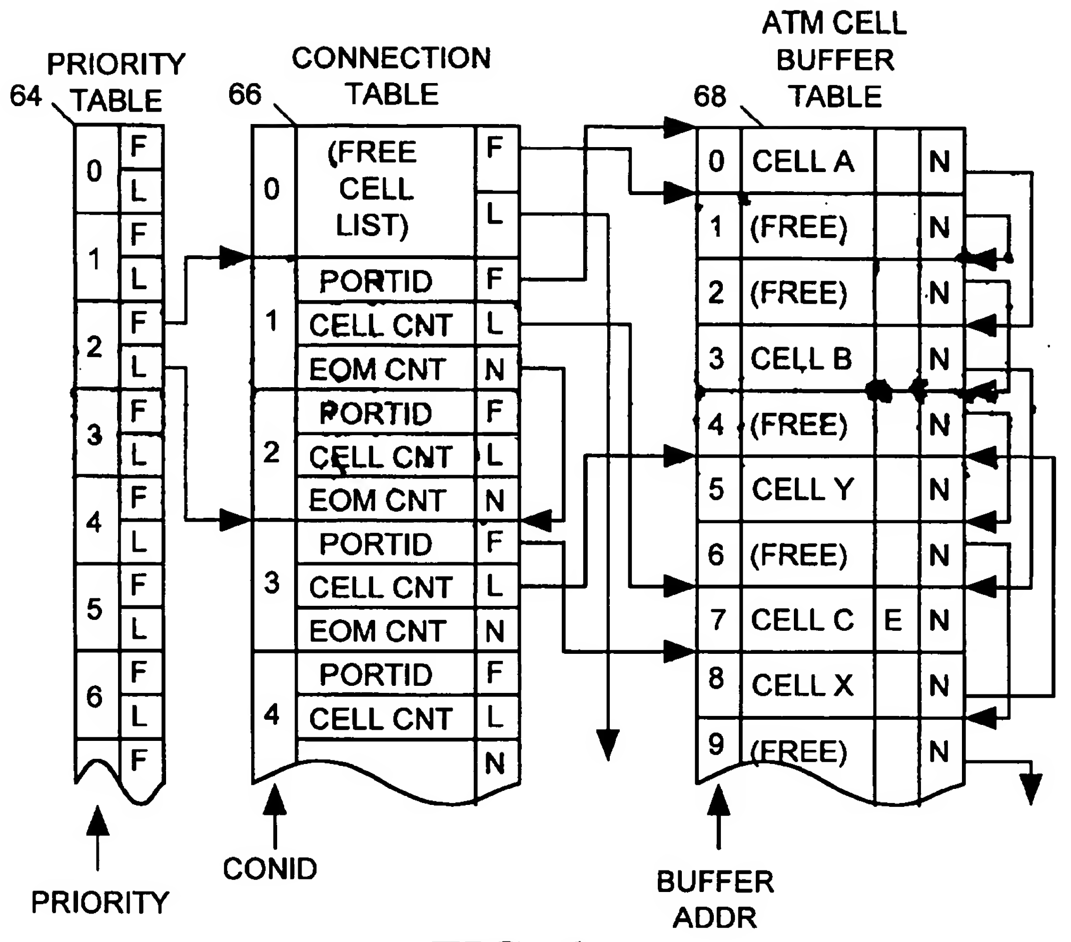

PRIORITY TABLE
64
CONNECTION TABLE
66
ATM CELL BUFFER TABLE
68
0 F L
1 F L
2 F L
3 F L
4 F L
5 F L
6 F L F
0 (FREE CELL LIST) F L
1 PORTID F
CELL CNT L
EOM CNT N
2 PORTID F
CELL CNT L
EOM CNT N
3 PORTID F
CELL CNT L
EOM CNT N
4 PORTID F
CELL CNT L
N
0 CELL A N
1 (FREE) N
2 (FREE) N
3 CELL B N
4 (FREE) N
5 CELL Y N
6 (FREE) N
7 CELL C E N
8 CELL X N
9 (FREE) N
PRIORITY
CONID
BUFFER ADDR

FIG. 6

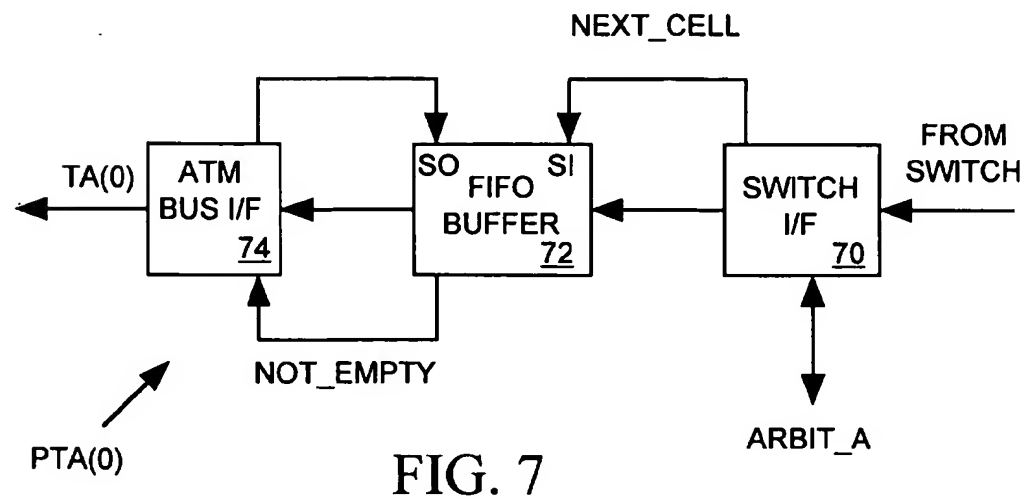

NEXT_CELL
TA(0)
ATM BUS I/F 74
SO
SI
FIFO BUFFER 72
SWITCH I/F 70
FROM SWITCH
NOT_EMPTY
ARBIT_A
PTA(0)

FIG. 7

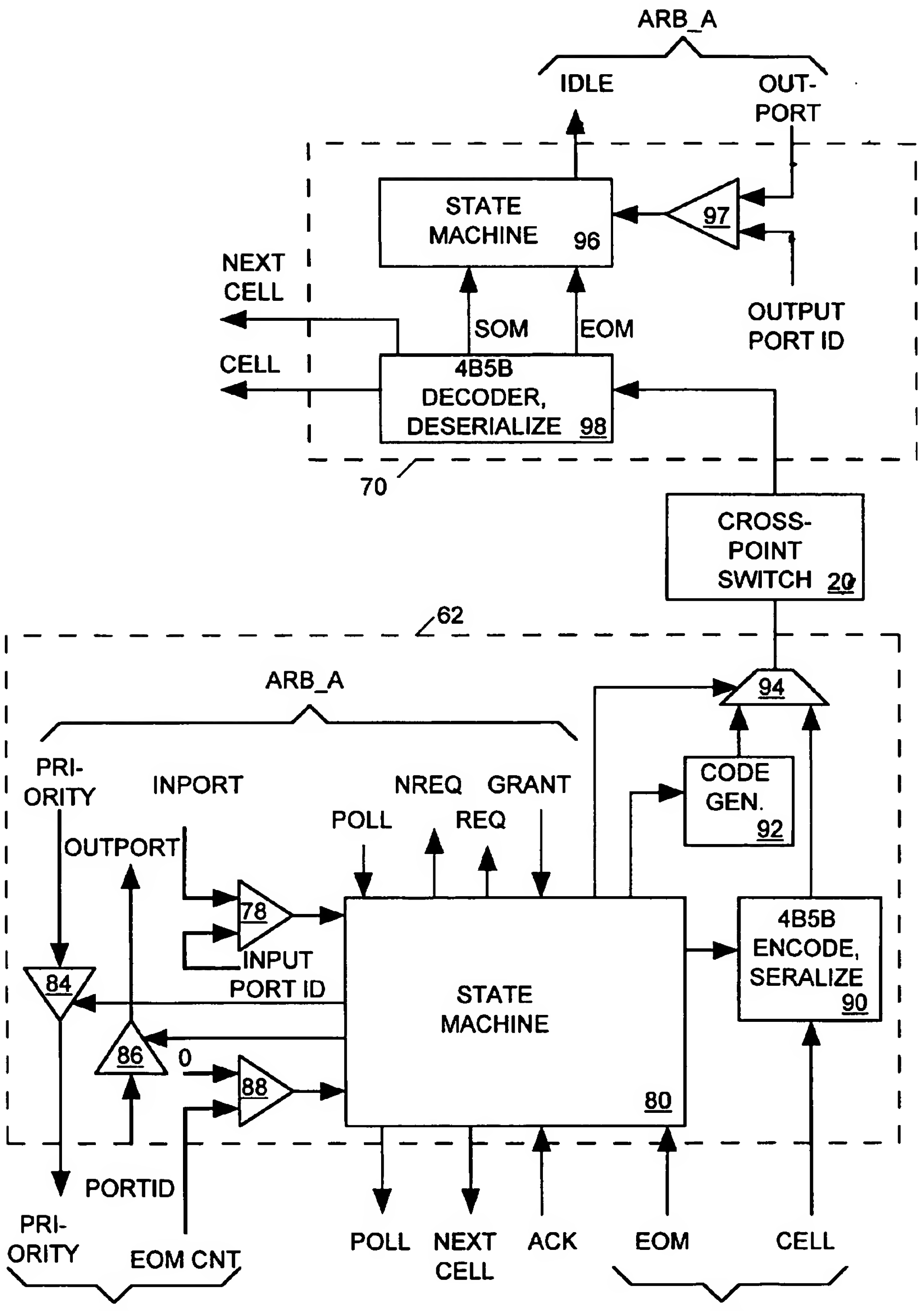

ARB_A
IDLE
OUT-PORT
STATE MACHINE 96
97
NEXT CELL
SOM
EOM
OUTPUT PORT ID
CELL
4B5B DECODER, DESERIALIZE 98
70
CROSS-POINT SWITCH 20
62
94
ARB_A
PRI-ORITY
INPORT
NREQ
GRANT
POLL
REQ
CODE GEN. 92
OUTPORT
78
INPUT PORT ID
4B5B ENCODE, SERALIZE 90
84
STATE MACHINE
86
0
88
80
PORTID
PRI-ORITY
EOM CNT
POLL
NEXT CELL
ACK
EOM
CELL
BUS 57
BUS 52

FIG. 8

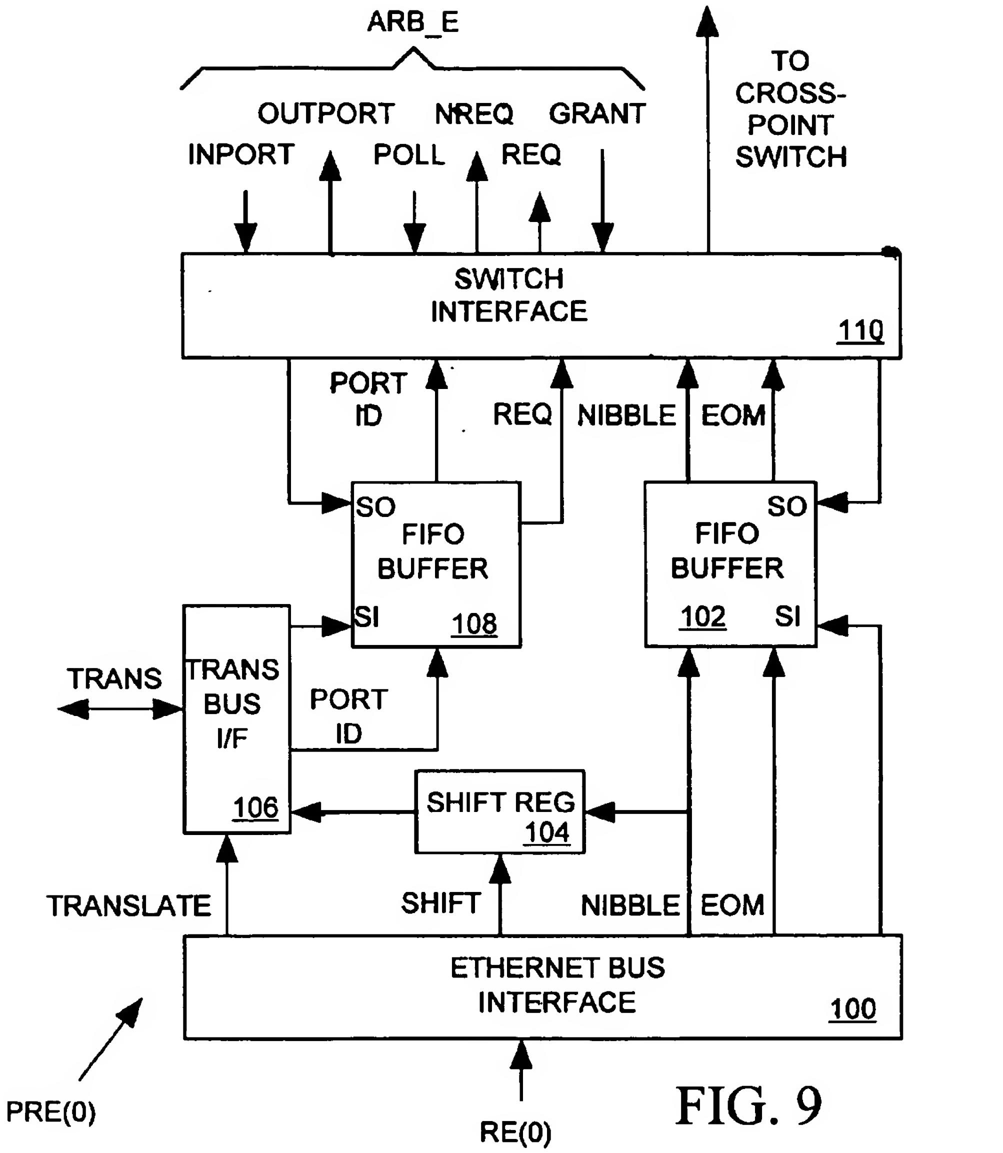

ARB_E
OUTPORT
NREQ
GRANT
INPORT
POLL
REQ
TO CROSS-POINT SWITCH
SWITCH INTERFACE
110
PORT ID
REQ
NIBBLE
EOM
SO
FIFO BUFFER
SI
108
SO
FIFO BUFFER
102
SI
TRANS
TRANS BUS I/F
106
PORT ID
SHIFT REG
104
TRANSLATE
SHIFT
NIBBLE
EOM
ETHERNET BUS INTERFACE
100
PRE(0)
RE(0)

FIG. 9

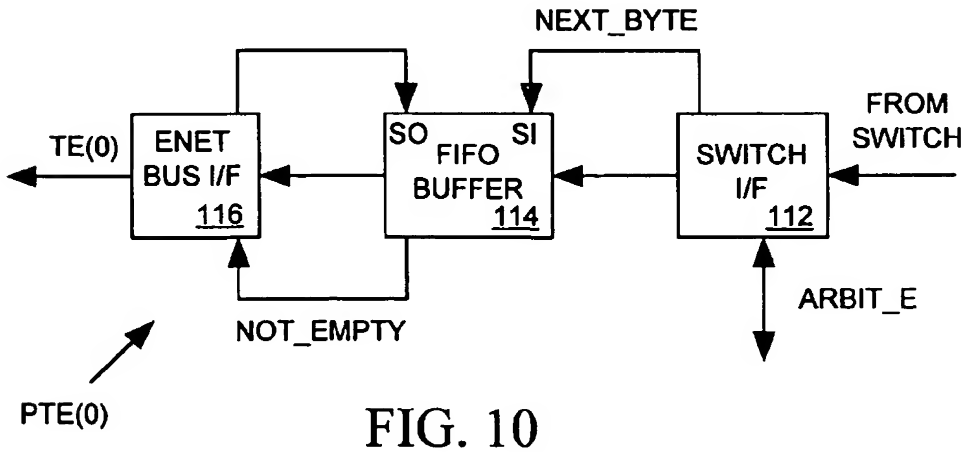

NEXT_BYTE
TE(0)
ENET BUS I/F
116
SO
SI
FIFO BUFFER
114
SWITCH I/F
112
FROM SWITCH
NOT_EMPTY
ARBIT_E
PTE(0)

FIG. 10

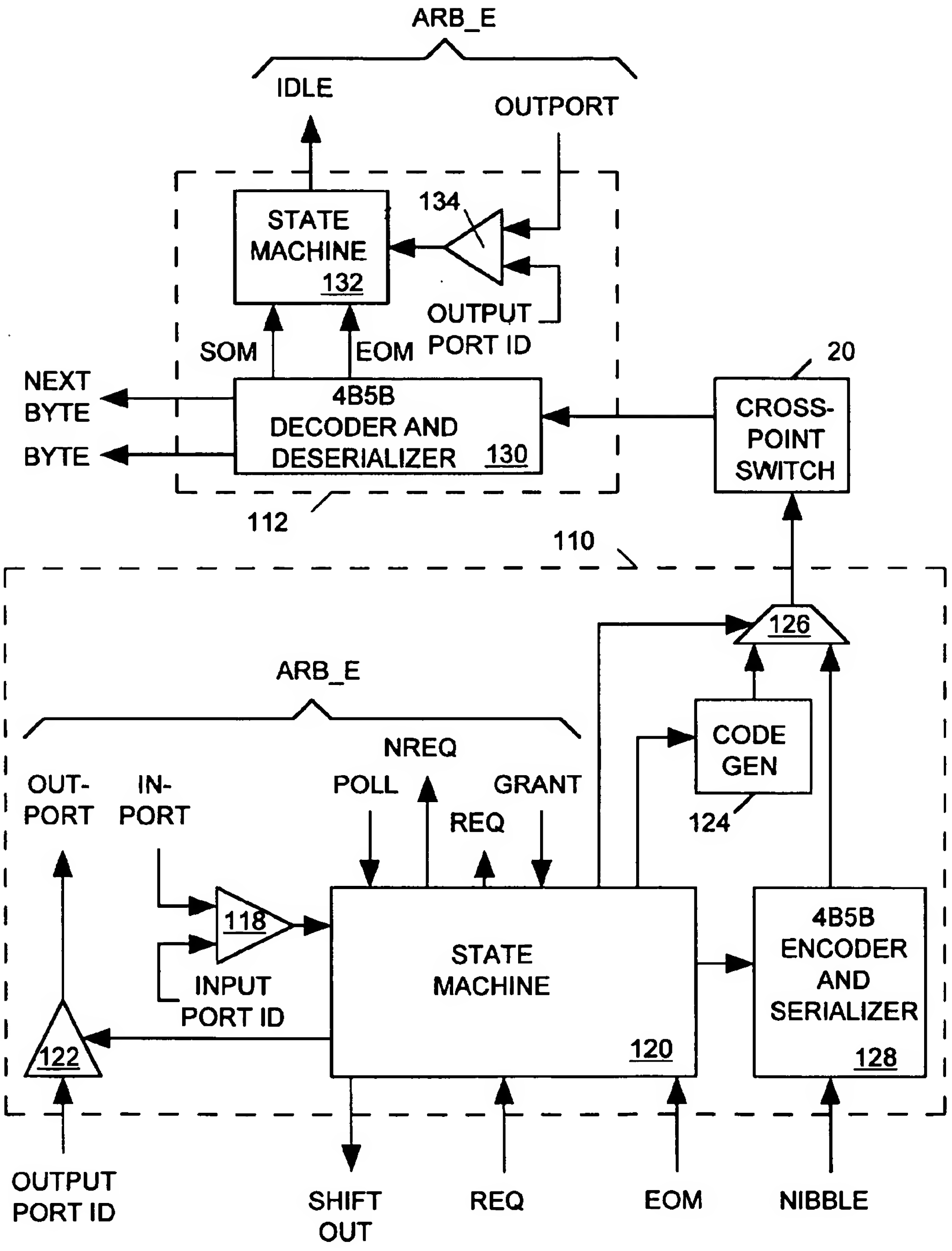

ARB_E
IDLE
OUTPORT
STATE MACHINE 132
134
OUTPUT PORT ID
SOM
EOM
NEXT BYTE
BYTE
4B5B DECODER AND DESERIALIZER 130
20
CROSS-POINT SWITCH
112
110
126
ARB_E
CODE GEN
124
NREQ
OUT-PORT
IN-PORT
POLL
REQ
GRANT
118
INPUT PORT ID
STATE MACHINE
120
4B5B ENCODER AND SERIALIZER
128
122
OUTPUT PORT ID
SHIFT OUT
REQ
EOM
NIBBLE

FIG. 11

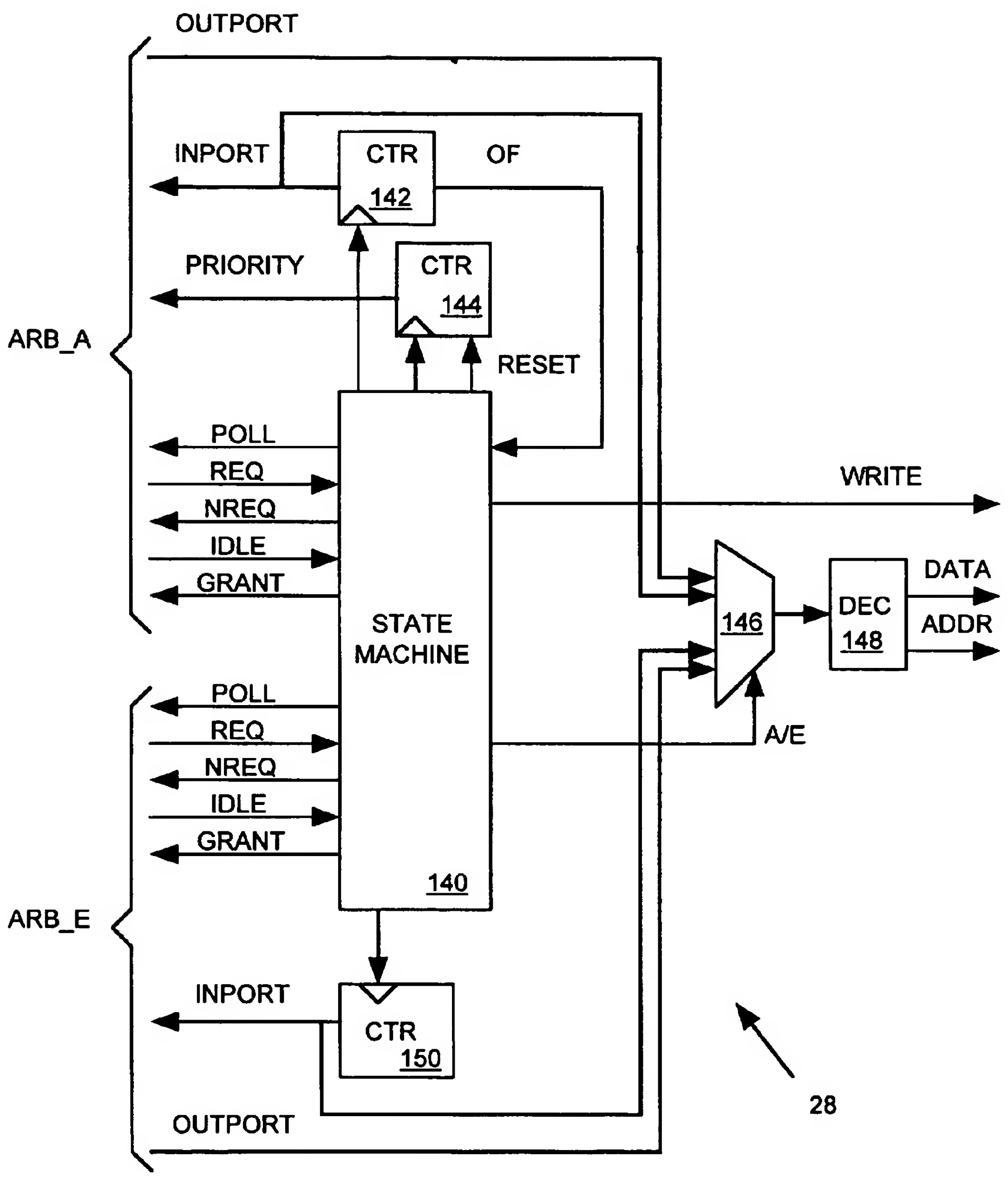

OUTPORT
INPORT
CTR
142
OF
PRIORITY
CTR
144
ARB_A
RESET
POLL
REQ
NREQ
IDLE
GRANT
WRITE
146
DEC
148
DATA
ADDR
STATE MACHINE
POLL
REQ
NREQ
IDLE
GRANT
A/E
140
ARB_E
INPORT
CTR
150
28
OUTPORT

FIG. 12

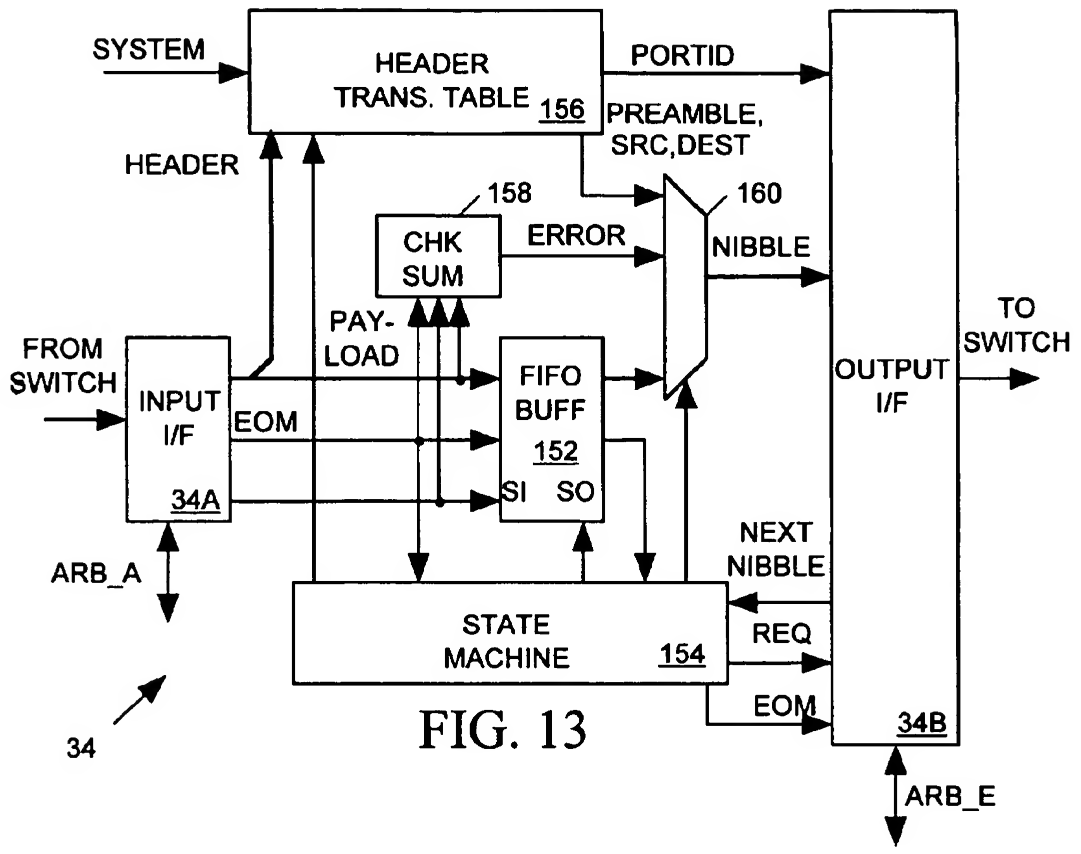

SYSTEM
HEADER TRANS. TABLE 156
PORTID
PREAMBLE, SRC,DEST
HEADER
158
160
CHK SUM
ERROR
NIBBLE
PAY-LOAD
TO SWITCH
FROM SWITCH
INPUT I/F
34A
EOM
FIFO BUFF 152
SI SO
OUTPUT I/F
ARB_A
NEXT NIBBLE
STATE MACHINE 154
REQ
EOM
34B
34
ARB_E

FIG. 13

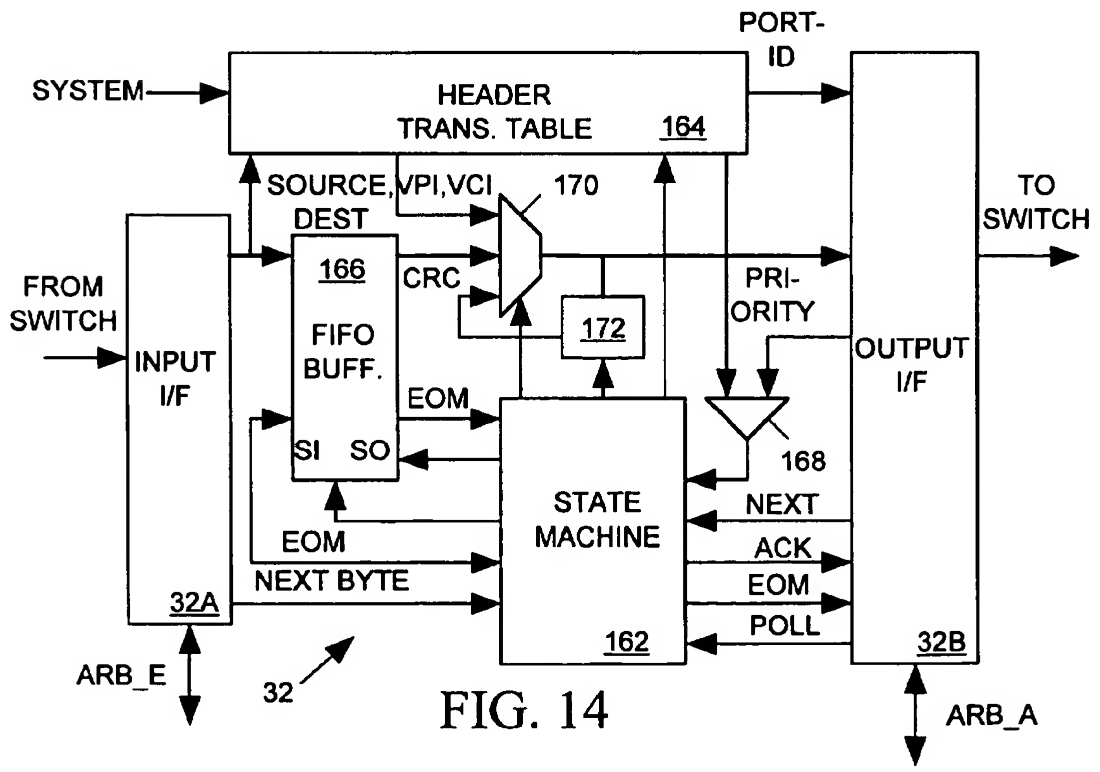

PORT-ID
SYSTEM
HEADER TRANS. TABLE 164
SOURCE, DEST
VPI,VCI
170
TO SWITCH
FROM SWITCH
166
FIFO BUFF.
CRC
172
PRI-ORITY
INPUT I/F
OUTPUT I/F
EOM
SI SO
168
STATE MACHINE 162
NEXT
EOM
ACK
NEXT BYTE
EOM
POLL
32A
32B
ARB_E
32
ARB_A

FIG. 14

NETWORK SWITCH PROVIDING PER VIRTUAL CHANNEL QUEUING FOR SEGMENTATION AND REASSEMBLY

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates in general to a network routing switch for routing both ATM and Ethernet protocol data transmissions between stations of a computer network, and in particular to a routing switch that translates between the two protocols.

2. Description of Related Art

Ethernet and asynchronous transfer mode (ATM) protocols are widely used to convey data between network stations. A station in an Ethernet network sends data to other stations in the form of a variable length Ethernet packet including, in addition to a payload of 46 to 1500 bytes of data, fields indicating its network source and destination addresses and other information. An Ethernet packet must be sent and received as a continuous transmission and cannot be interleaved with other transmissions.

A station in an ATM network segments a data transmission into a variable number of 48-bit ATM cells. Each ATM cell includes a fixed length data payload, a virtual path identifier (VPI), a virtual channel identifier (VCI) and other information. All ATM cells forming an ATM transmission have the same VCI and VPI identifiers. The VPI field identifies a particular virtual path that the cells of the transmission are to follow through the network. Each virtual path may have several virtual channels. The VCI field identifies the particular virtual channel within a virtual path to which a transmission is assigned. When two ATM transmissions are assigned to the same virtual channel of the same virtual path, one transmission must follow the other; their cells may not be interleaved. An end of message (EOM) code included in the last cell of each transmission marks the boundary between that transmission and the next transmission of the same virtual path and channel. Cells of several ATM transmissions assigned to the same virtual path may be interleaved on the same physical transmission line at any point along that path provided they are assigned to different virtual channels. The VPI field in the cells allows a receiving network device to sort them by transmission.

In one respect Ethernet packets are more efficient than ATM transmissions because an Ethernet packet can contain more payload data per unit of routing information than an ATM cell. However in networks where stations must compete for high speed transmission channels, the ability to interleaved ATM cells allows more efficient use of data transmission bandwidth.

It is sometimes advantageous for a network to employ both ATM and Ethernet protocols. For example a small cluster of closely associated network stations interconnected by a network routing switch may communicate with one another via Ethernet protocol. However when routing switches interconnecting such clusters of network stations are themselves interconnected by high speed channels, it may be advantageous to employ ATM protocol when sending data over those high speed channels. In such systems "segmentation" and "reassembly" units are used to convert between Ethernet and ATM transmission protocols. A segmentation unit segments a large Ethernet packet into a sequence of smaller ATM cells. A reassembly unit "reassembles" a transmission of small ATM cells into one or more large Ethernet packets.

Prior art routing switches handle only one transmission protocol. Thus, for example, when a network switch uses

Ethernet protocol, a reassembly unit must convert each ATM transmission to Ethernet protocol before the transmission arrives at the switch. Conversely a segmentation unit must convert an Ethernet packet emerging from the switch to ATM protocol before it can be forwarded via a high speed ATM channel. Thus segmentation and reassembly units are required at each Ethernet switch port that is to communicate via ATM protocol.

A segmentation unit is relatively simple and requires relatively little storage capacity because it need only process one Ethernet packet at a time. However since cells of many (perhaps hundreds) of different ATM transmissions may arrive at a reassembly unit interleaved, a reassembly unit requires much logic and memory to sort out the interleaved ATM transmissions and to store the cells of each ATM transmission until they can be assembled into an Ethernet packet. Thus each reassembly unit requires access to a large memory for storing ATM cells of large numbers of data transmissions and also needs complicated logic for keeping track of the ATM transmissions that it is storing. Due to the complicated nature of the reassembly unit and its need for a large, external memory, a segmentation/reassembly unit capable of handling high speed data traffic is physically large and requires numerous I/O pins. Since such a segmentation/reassembly unit may be needed for each of several switch ports, a network switch capable of ATM/Ethernet translation is large and expensive.

Also since the routing switch operates only in one protocol, an ATM packet destined for an ATM system must first be translated into Ethernet protocol, sent through the Ethernet protocol routing switch, and then converted back into ATM protocol. The double translation of ATM packets simply to move them through an Ethernet protocol switch reduces system throughput.

What is needed is a network switch that routes both ATM and Ethernet protocol transmissions, that provides segmentation and reassembly only when necessary, and which does not employ a large number of complicated segmentation/reassembly units.

SUMMARY OF THE INVENTION

A network routing switch in accordance with the present invention includes a crosspoint switch, a set of Ethernet I/O ports, a set of ATM I/O ports, a reassembly unit for converting ATM transmissions into Ethernet transmissions, and a segmentation unit for converting Ethernet transmissions into ATM transmissions. As an ATM I/O port receives cells of an ATM transmission from an external source it stores them until the transmission is complete and then sends the ATM transmission through the crosspoint switch either to a forwarding ATM port or to the reassembly unit depending on whether the ATM transmission is to be forwarded as an ATM or Ethernet transmission. When the reassembly unit receives an ATM transmission it converts it to an Ethernet transmission and forwards it through the crosspoint switch to a forwarding Ethernet I/O port. When an Ethernet I/O port receives an Ethernet packet from an external source it sends the packet through the crosspoint switch either to a forwarding Ethernet port or to the segmentation unit depending on whether the Ethernet packet is to be forwarded as an Ethernet or ATM transmission. When the segmentation unit receives an Ethernet packet it converts it to an ATM transmission and forwards it through the crosspoint switch to a forwarding ATM I/O port.

It is accordingly an object of the invention to provide a network switch that can route both ATM and Ethernet

packets and which translates between ATM and Ethernet protocols only when necessary.

It is another object of the invention to provide a network switch that can translate between ATM and Ethernet packets using only a single segmentation unit and a single reassembly unit, rather than a segmentation and reassembly unit at each of several I/O ports.

The concluding portion of this specification particularly points out and distinctly claims the subject matter of the present invention. However those skilled in the art will best understand both the organization and method of operation of the invention, together with further advantages and objects thereof, by reading the remaining portions of the specification in view of the accompanying drawing(s) wherein like reference characters refer to like elements.

BRIEF DESCRIPTION OF THE DRAWING(S)

FIG. 1 illustrates a computer network employing a set of network routing switches in accordance with the present invention;

FIG. 2 illustrates one of the routing switches of FIG. 1 in more detailed block diagram form;

FIG. 3 illustrates a typical routing switch of FIGS. 1 and 2 in more detailed block diagram form;

FIG. 4A illustrates the structure of an ATM cell;

FIG. 4B illustrates the structure of an Ethernet packet;

FIG. 5 illustrates an ATM input port of FIG. 3 in more detailed block diagram form;

FIG. 6 illustrates the data structures maintained by the input port of FIG. 5;

FIG. 7 illustrates the ATM output port of FIG. 3 in more detailed block diagram form;

FIG. 8 illustrates the ATM output port switch interface circuit of FIG. 5 and the ATM input port switch interface circuit of FIG. 7 in more detailed block diagram form;

FIG. 9 illustrates the Ethernet input port of FIG. 3 in more detailed block diagram form;

FIG. 10 illustrates the Ethernet output port of FIG. 3 in more detailed block diagram form;

FIG. 11 illustrates the input port switch interface of FIG. 9 and the output port switch interface of FIG. 10 in more detailed block diagram form;

FIG. 12 illustrates the arbiter of FIG. 3 in more detailed block diagram form;

FIG. 13 illustrates the reassembly unit of FIG. 2 in more detailed block diagram form; and

FIG. 14 illustrates the segmentation unit of FIG. 2 in more detailed block diagram form.

DESCRIPTION OF THE PREFERRED EMBODIMENT(S)

FIG. 1 illustrates a computer network 10 employing a set of network routing switches 12 in accordance with the present invention. Switches 12 interconnect network stations 14 employing Ethernet protocol and networks stations 16 employing ATM protocol and provide the protocol translation when necessary to allow stations 14 and 16 to communicate with each other. Routing switches 12 are themselves linked either by high speed Ethernet communication paths 18 or ATM communication paths 20.

When an Ethernet station 14 transmits an Ethernet packet to a switch 12, the switch determines from routing data included in the transmission whether it must forward the

transmission to a local Ethernet station 14, a local ATM station 16, or to another switch 12 via an Ethernet path 18 or an ATM path 20. If the Ethernet packet is to be forwarded to a local Ethernet station 14 or to a remote switch 12 via an Ethernet path 18, the receiving switch 12 forwards the Ethernet packet without translation. However if the Ethernet packet is to be forwarded to a local ATM station 16 or to a remote switch 12 via an ATM path 20, the receiving switch 12 first translates the Ethernet protocol transmission to an ATM protocol transmission before forwarding it to its destination. Conversely, when a switch 12 receives an ATM transmission it translates the ATM transmission to one or more Ethernet packets only when it is to be forwarded to a local Ethernet station 14 or to a remote switch 12 via an Ethernet path 18.

Routing Switch Architecture

FIG. 2 illustrates one of the routing switches 12 of FIG. 1 in simplified block diagram form. Routing switch 12 includes a crosspoint switch 20 interconnecting a set of Ethernet I/O ports 24, and a set of ATM I/O ports 26. Routing switch 12 also includes a segmentation unit 32 for converting Ethernet packets into ATM transmissions and a reassembly unit 34 for converting ATM transmissions into Ethernet packets. Crosspoint switch 20 provides communication paths between I/O ports 24 and 26 and segmentation and reassembly units 32,34 in response to control data from a routing arbiter 28.

When an ATM port 26 receives an incoming ATM transmission via an ATM bus, it stores each cell of the transmission in an internal memory. Although cells of separate ATM transmissions may arrive interleaved at an ATM port 26, the port looks at data included in each incoming cell to determine the transmission to which it belongs and stores the ATM cell accordingly. The ATM port 26 also determines from information included in the ATM cells of each transmission how the transmission is to be forwarded through switch 20. If the ATM transmission is to be forwarded through another ATM port 26, the receiving ATM port 26 requests arbitrator 28 to route the transmission directly to the forwarding ATM port 26. When the forwarding ATM port 26 is idle (not busy forwarding another transmission), arbitrator 28 establishes a path between the receiving and forwarding ATM ports 26 through switch 20 and signals the receiving ATM port 26 to begin forwarding the ATM transmission to the forwarding ATM port 26.

On the other hand, if an ATM port 26 determines that an incoming ATM transmission is to be forwarded outward from switch 12 in the form of one or more Ethernet packets, the receiving ATM port 26 requests arbitrator 28 to route the ATM transmission to reassembly unit 34. When the reassembly unit 34 is ready to receive the ATM transmission, arbitrator 28 establishes a path through switch 20 between the receiving ATM port 26 and reassembly unit 34 and signals the receiving ATM port to begin forwarding the ATM transmission to the reassembly unit. Reassembly unit 34 then converts the ATM transmission to one or more Ethernet packets, determines which Ethernet port 24 is to forward the Ethernet packet(s), and then requests arbitrator 28 to route the Ethernet packet(s) through switch 20 to the forwarding Ethernet port 24.

When an Ethernet port 24 receives an incoming Ethernet packet it stores the packet and determines whether the data conveyed in the packet is to be forwarded outward from switch IL via another Ethernet port 24 via an ATM port 26. When the packet is to be forwarded by an Ethernet port, the receiving Ethernet port 24 requests arbitrator 28 to establish a path through switch 20 to the forwarding Ethernet port 24, and forwards the Ethernet packet when the path is established. When an Ethernet port 24 receives an Ethernet packet that is to be forwarded as an ATM transmission, the Ethernet port 24 requests arbitrator 28 to route the Ethernet packet through switch 20 to segmentation unit 32. Segmentation unit 32 then converts the Ethernet packet to an ATM transmission, determines which ATM port 26 is to forward the ATM transmission, and requests arbitrator 28 to route the ATM transmission to the forwarding ATM port 26.

FIG. 3 illustrates routing switch 12 of FIGS. 1 and 2 in more detailed block diagram form. Switch 12 includes N ATM I/O ports and M Ethernet I/O ports. Each ATM I/O port is shown in FIG. 3 as including two components: one of ATM input ports PRA(0)–PRA(N) and a corresponding one ATM output ports PTA(0)–PTA(N). Each of the M Ethernet I/O ports is also shown as including two components: one of Ethernet input ports PRE(0)–PRE(M) and a corresponding one of Ethernet output ports PTE(0)–PTE(M). Incoming ATM transmissions arrive at input ports PRA(O)–PRA(N) via corresponding transmission lines RA(0)–RA(N) and incoming Ethernet packets arrive at input ports PRE(0)–PRE (M) via corresponding transmission lines RE(0)–RE(M). Outgoing ATM transmissions depart from output ports PTA (0)–PTA(N) via corresponding transmission lines TA(0)–TA (N) and outgoing Ethernet packets depart from output ports PTE(0)–PTE(M) via corresponding transmission lines TE(0)–TE(M). Segmentation unit 32 includes three components: an input interface 32A, an output interface 32B, and a processing section (not shown in FIG. 3) connected between its input and output interfaces. Similarly reassembly unit 34 also includes three components: an input interface 34A and an output interface 34B and a processing section (not shown in FIG. 3) connected between its input and output interfaces.

Crosspoint switch 20 includes a set of "vertical" conductors 45 connected to input ports PRA(0)–PRA(N) and PRE (0)–PRE(N) and to segmentation and reassembly unit output interfaces 32B and 34B. Switch 20 also includes a set of "horizontal" conductors 47 connected to input ports PRA (0)–PRA(N) and PRE(0)–PRE(N) and to the segmentation and reassembly unit input interfaces 32A and 34A. When turned on, a pass transistor 48 at each crosspoint between a horizontal conductor 47 and a vertical conductor 45 forms a signal path between the two conductors. Each bit of an M+N+2 bit data word at each address of a random access memory (RAM) 42 controls the on/off state of a separate one of the M+N+2 transistors 48 along one horizontal row of switch 20. To connect a particular input port to a particular output port, arbiter 28 writes one data word to RAM 42. That data word turns on one pass transistor 48 and turns off all others along the horizontal line 47 leading to the output port.

Arbitrator 28 communicates with all ATM input and output ports PRA(0)–PRA(N) and PTA(0)–PTA(N), the reassembly unit input interface 34A and the segmentation unit output interface 32B through a parallel arbitration bus ARB_A. Using the ARB_A bus, arbitrator 28 polls the ATM input and output ports to determine when and where an input port wants to send an ATM transmission and to determine whether an ATM output port is ready to receive and forward an ATM transmission. Arbiter 28 also uses the ARB_A bus to poll segmentation unit output interface 32B to determine when and where it wants to send an ATM transmission it has created and to poll reassembly unit input interface 34A to determine when it is ready to receive and reassemble an ATM transmission. The information obtained

tells arbiter 28 how and when to route ATM transmissions through switch 20.

In a similar manner arbitrator 28 uses another parallel arbitration bus ARB_E to poll all Ethernet input and output ports PRE(0)–PRE(M) and PTE(0)–PTE(M), segmentation unit input interface 32A and reassembly unit output interface 34B in order to determine how and when to route Ethernet transmissions through switch 20.

ATM Cell Format and Header Translation

FIG. 4A illustrates the structure of an ATM cell. The fixed length ATM cell includes a virtual path identifier (VPI), a virtual channel identifier (VCI), an error check field (CRC) and a data field (PAYLOAD). Each ATM cell is a part of an ATM transmission made up of a sequence of such ATM cells and all ATM cells forming an ATM transmission have the same VCI and VPI identifiers. The VPI field identifies a particular virtual path that the cells of the transmission is to follow through the network. The VCI field identifies a particular virtual channel of that path. Cells of several ATM transmissions assigned to the same virtual path may be interleaved if they are assigned to differing virtual channels because the VPI field in the cells distinguishes them by transmission. However, two ATM transmissions are assigned to the same virtual channel of the same virtual path, one transmission must follow the other; their cells may not be interleaved. An end of message (EOM) code included in the PAYLOAD field of the last cell of each transmission marks the boundary between that transmission and the next.

An ATM transmission can be routed from an ATM input port PRA(0)–PRA(N) or from segmentation unit output interface 32B either to one of ATM output ports PTA(0) –PTA(N) or to the reassembly unit input interface 34A. Each ATM and Ethernet input/out port PRA/PTA and PRE/PTE has a unique switch port ID. Segmentation and reassembly units 32,34 also have unique switch port IDs. Each ATM input port PRA(0)–PRA(N) and segmentation unit output interface 32B includes an internal lookup table relating each VPI value to the port ID of an ATM output port PTA(0) –PTA(N) or reassembly input unit 34B to which an ATM transmission having that VCI code is to be forwarded. The lookup table also indicates a relative forwarding "priority" for each combination of VCI and VPI. When an ATM input port PRA(0)–PRA(N) or the segmentation unit 34 is ready to forward an ATM transmission, it competes for a connection to the destination ATM output port PTA(0)–PTA(N) or to reassembly input unit 34A. Arbitrator 28 grants connections in order of forwarding priority. The RAM-based lookup tables in the ATM output ports and in segmentation output unit 32B are maintained by an external host supplying translation data via a conventional memory bus (SYSTEM).

Ethernet Packet Format and Address Translation

FIG. 4B illustrates the structure of a standard Ethernet packet. An Ethernet packet includes a preamble, a start of frame delimiter (SOF), a destination address (DEST), a source address (SOURCE), a type field (TYPE), a data field (DATA) and an error check field (ERROR). All fields except the DATA field are fixed in length; the DATA field can range between 46 and 1500 bytes. Each Ethernet station in a network is identified by a unique network address. The destination address DEST contained in the packet indicates the network address of the station to receive the packet. The SOURCE address included in the packet indicates the station that sent the packet.

When an Ethernet input port PRE(0)–PRE(M) receives an incoming packet, it sends its own port ID, along with the packet's SOURCE and DEST fields, to an address translation system 44 (FIG. 3) via a parallel bus TRANS. Translation system 44 maintains a lookup table relating each network address of each network station to the switch port ID of an Ethernet I/O port that communicates with that station either directly or through another network switch. Address translation system 44 uses the incoming port ID and source network address to update its lookup table. It also responds to the incoming destination network address by returning the port ID of the switch output port PRE(0)–PTE (M) that can forward an Ethernet packet to the network station identified by the destination address. Later, when it is ready to send the packet onward, the Ethernet input port PRE(0)–PRE(M) arbitrates for the right to forward the packet to the indicated Ethernet output port. When reassembly unit 34 converts an ATM transmission into an Ethernet packet, it also accesses address translation unit 44 via the TRANS bus to obtain the port ID of the Ethernet output port PTE(0)–PTE(M) to receive and forward the packet.

Address translation system 44 distinguishes Ethernet ports from ATM ports. When an Ethernet input port PRE (0)–PRE(M) has received a packet that must be converted to an ATM transmission, and sends the packet's destination address to address translation system 44, the address transmission system returns the switch port ID of the segmentation input unit 32A instead of a switch port ID of a switch port associated with the packet's destination address. This causes the Ethernet input port PRE(0)–PRE(M) that received the packet to request arbitrator 28 to forward the packet to segmentation unit 32 where it can be converted into an ATM transmission. The external host writes the data into translation table 44 via the SYSTEM bus that is needed to relate a network destination address to the segmentation unit.

ATM Input Port Architecture

FIG. 5 illustrates ATM input port PRA(0) of FIG. 3 in more detailed block diagram form. Other ATM input ports PR(1)–PR(N) are similar. ATM input port PRA(0) includes a conventional ATM bus interface circuit 50 which receives each ATM cell arriving on line RA(0), converts it to parallel form, and stores it in an internal FIFO buffer. When ready to forward a cell out of its FIFO buffer, bus interface circuit 50 sends a NEXT CELL signal to a queue manager 52. When ready to process the cell, queue manager 54 sends an acknowledge (ACK) signal to bus interface circuit 50. Interface circuit 50 then places the cell on a bus 52. ATM bus interface 50 also parses the payload of the ATM cell to determine if it contains the end of message (EOM) code. If it does, bus interface 50 sends an EOM signal to queue manager 54 and places an EOM bit on a line of bus 52. The VPI and VCI fields of the ATM cell on bus 52 provide input to a RAM-based lookup table 56 relating the VPI and VCI data values to a unique connection ID (CONID), a priority level (PRIORITY) and the ID (PORTID) of an output port or reassembly unit to receive the ATM transmission. In response to a signal from queue manager 54, lookup table 56 places its output PORTID, CONID, and PRIORITY data on lines of a bus 57.

Queue manager 54 stores an ATM cell appearing on bus 52 in a cell buffer RAM 58. A set of RAMs 60 implement lookup tables which keep track of where in RAM 58 the ATM cells for each transmission are stored and the priority of each transmission. Queue manager 54 read accesses the tables in RAMs 60 based on the CONID and PRIORITY

data appearing on bus 57 to determine an address within cell buffer RAM 58 at which an incoming ATM cell on bus 52 is to be stored. When one of RAMs 60 reads that address onto address lines of bus 52 queue manager 54 sends a write signal to RAM 58 causing it to store the ATM cell and the EOM bit, if any, appearing on bus 52 at that address. Queue manager 54 then updates tables in RAMs 60 to record the storage location of that ATM cell and to record the PORTID and PRIORITY levels associated with its transmission.

ATM input port PRA(0) includes a switch interface circuit 62 providing a link between the input port to a vertical line 45 of crosspoint switch 20 of FIG. 3. Switch interface circuit 62 also provides a communication link between the ATM arbitration bus ARB_A and queue manager 54. Arbiter 28 of FIG. 3 periodically polls each ATM input port to determine if it is ready to forward a complete ATM transmission of a particular priority to an ATM output port or to the segmentation unit. To poll an input port, the arbiter 28 places the input port's ID on INPORT lines of the ARB_A bus and pulses a POLL signal. It also places a priority code on PRIORITY lines of the ARB_A bus. If it is not busy forwarding an ATM transmission to an output port, switch interface circuit 62 of the input port having the port ID appearing on the INPORT lines places the priority value on lines of bus 57 and forwards the POLL signal to queue manager 54. Queue manager 54 then write accesses RAMs 60 to determine whether the port is ready to forward a transmission having that priority. If so, queue manager 54 causes RAM 60 to read out the port ID of the output port (or reassembly unit) that is to receive the transmission onto lines of bus 57. Switch interface circuit 62 puts the output port ID on OUTPORT lines of the ARB_BUS and asserts an ARB_A bus request line (REQ). Each output port monitors the OUTPORT and REQ lines and in response to the REQ line, the output port having the ID appearing on the OUTPORT lines signals arbiter 28 (FIG. 3) as to whether it is idle or busy receiving a transmission. If the requested output port is idle, the arbiter establishes a connection between the requesting input port and the requested output port and pulses a GRANT signal line of the ARB_A bus. Switch interface circuit 62 responds to the GRANT signal by pulsing a NEXT CELL signal to queue manager 54. Queue manager 54 responds to each NEXT CELL signal pulse by causing RAM 58 to read out an ATM cell of the ATM transmission onto bus 52 to forward the cell to switch interface 62. Queue manager then sends an acknowledge signal [ACK] to switch interface circuit 62. Switch interface circuit 62 then serializes, 4B5B encodes and forwards the cell to the output port via vertical line 45 of the crosspoint switch. (4B5B encoding is discussed below.) When queue manager 54 sends the last cell of the transmission to switch interface 62, an EOM bit appearing with the last ATM cell on bus 52 tells switch interface 62 that the input port is to be idle. After forwarding the last cell, switch interface 62 awaits the next poll from the arbiter.

ATM Input Port Data Structures

FIG. 6 illustrates the data structures stored in RAMs 58 and 60 of input port PRA(0) of FIG. 5. RAM 58 acts as an ATM cell buffer table 68, storing one ATM cell at each address. Each ATM cell buffer containing an ATM cell also stores a next cell pointer "N" to a cell buffer containing a next ATM cell of the same transmission items. Thus, the cell buffers use their N pointers to form linked lists. Each linked list corresponds to a separate virtual channel and contains all of the ATM cells assigned to that virtual channel in the order they arrive at the port. Each cell buffer stores a bit "E" set when the cell buffer contains such an EOM cell. All ATM cell buffer locations not currently storing an ATM cell form a separate "free cell linked list". The CONID output at RAM **56** references a particular linked list to which the incoming cell is to be added. A connection table **66** implemented by one of RAMs **60** addressed by the CONID data contains an entry for the free cell liked list and one entry for each linked list in the ATM cell buffer table **68**. Each connection table entry includes a pointer "F" to the first cell buffer of the corresponding linked list, a pointer "L" to the last cell buffer of the linked list, the ID (PORTID) of the output port to which the ATM transmission is to be forwarded, a data value "Cell CNT" indicating the number of cells in the linked list, and data value EOM CNT indicating the number of complete ATM transmissions included in the linked list.

Connection table **66** entries are also organized into linked lists, one for each possible forwarding priority level. All connections having the same priority are linked by the same connection table list. A priority table **64** implemented by one of RAMs **60** addressed by a PRIORITY data value includes an entry for each priority level. Each priority table entry includes pointers F and L to the first and last entry of the connection table **66** linked list for the corresponding priority level. The manner in which queue manager **54** of FIG. **5** uses and maintains the tables illustrated in FIG. **6** is discussed below.

ATM Output Port Architecture

FIG. **7** illustrates ATM output port PTA(**0**) of FIG. **3** in more detailed block diagram form. Output ports PTA(**1**) –PTA(N) are similar. Output port PTA(**0**) includes a switch interface circuit **50** for receiving serial transmissions of 4B5B encoded, serialized ATM cells from crosspoint switch **20** of FIG. **3**. Switch interface circuit **70** 4B5B decodes and deserializes each cell and stores it in a first-in, first-out (FIFO) buffer **72** by pulsing a shift-in(SI) input of buffer **72**. When not empty, FIFO buffer **72** sends a NOT_EMPTY signal to an ATM bus interface circuit **74**. ATM bus interface circuit **74** responds by strobing a shift out (SO) FIFO buffer **52** control input causing the FIFO buffer to shift out a longest stored ATM cell. ATM bus interface circuit **74** then serializes the ATM cell and transmits it outward on ATM bus TA(**0**) using ATM protocol. As discussed below, arbiter **28** of FIG. **3** polls switch interface circuit **70** via the ARBIT_A bus to determine when output port PTA(**0**) is idle.

ATM Arbitration

FIG. **8** illustrates ATM output port switch interface circuit **62** of FIG. **5** and ATM input port switch interface circuit **70** of FIG. **7** in more detailed block diagram form. To poll an ATM input port, arbiter **28** of FIG. **3** places a priority level value and input port ID on the PRIORITY and INPORT lines of the ARB_A bus and pulses the POLL line of the ARB_A bus. A comparator **78** compares the input port's PORT ID with the input port ID on the INPUT lines and signals a state machine **80** when they match. When the port is busy forwarding an ATM transmission to an output port, state machine **80** pulses an NREQ line of the ARB_A bus to indicate that it does not want to request a connection to an output port. The arbiter then polls another input port. When the polled input port **62** is not busy forwarding an ATM transmission to an output port, a state machine **80** responds to the signal from comparator **78** and the POLL signal pulse by forwarding the PRIORITY line data via a buffer **82** to lines of bus **57** via a buffer **84**, by connecting other lines of bus **57** to the OUTPORT lines of the ARB_A bus via a buffer **86**, and by forwarding the POLL signal to input port queue manager **54** of FIG. **5**.

Using the PRIORITY data (for example Priority=z) appearing on the ARB_A bus to address the priority table **64** of FIG. **6**, queue manager **54** causes the first connection table **66** entry for that priority (e.g. entry **1**). If the input ports stores no completed ATM transmissions at that priority level, the EOM count will be 0. A comparator **88** in interface circuit **62** compares the EOM count on bus **57** to 0 and signals state machine **80** accordingly. If the EOM count is 0, state machine **80** pulses the NREQ line to indicate that is not requesting a connection. If the EOM count is greater than 0, state machine **80** causes buffer **86** to put the output port ID readout of table **66** on the OUTPORT lines of the ARB_A bus and pulses the REQ line of the ARB_A bus to request a connection to an output port. If the output port referenced by the port ID now appearing on the OUTPORT lines is idle, it pulses an IDLE line of the ARB_A bus causing the arbiter to establish a connection through crosspoint switch **20** to the referenced output port.

Upon establishing the connection, the arbiter pulses a GRANT signal line of the ARB_A bus. State machine **80** responds by signaling a code generator circuit **92** to transmit a 4B5B encoded "start of transmission" code sequence via a multiplexer **94** through crosspoint switch **20** to the output port. When the output port detects this sequence it prepares to receive the ATM transmission. State machine **80** then sends a NEXT_CELL signal to queue manager **54** causing queue manager **54** to send the first cell of the stored ATM transmission outward on bus **52** to a circuit **90**. Queue manager **54** then pulses an acknowledge signal (ACK) input to state machine **80**. State machine **80** than signals circuit **90** to serialize the cell and convert it to the well-known "4B5B" encoded form. In this form of encoding, each 4-bit nibble of the cell is converted into a 5-bit code. While many 5-bit 4B5B codes correspond to encoded 4-bit values, some 5-bit 4B5B code values are reserved for use as various synchronizing codes, start and end of message codes and the like. State machine **80** forwards each encoded ATM cell to output port switch interface circuit **70** via multiplexer **94** and switch **20**. Thereafter, queue manager **54** sends each cell of the ATM transmission to circuit **90** in response to each NEXT CELL signal pulse and state machine **80** responds to each ACK signal pulse by forwarding the encoded serialized cell from circuit **90** to the interface circuit **70** of the destination output port. When the last CELL of the transmission appears on BUS **52** the EOM bit stored with that cell appears on a line of bus **52** providing input to state machine **80**. After forwarding that cell to the output port, state machine **80** signals code generator **92** and multiplexer **94** to send an end of transmission code to the output port. The EOM bit also tells queue manager **54** to decrement the EOM count in the connection table entry to update the free cell list and to remove the freed ATM cell buffer table entries from the cell buffer linked list.

Output port switch interface circuit **70** of FIG. **7**, shown in more detail in FIG. **8**, includes a state machine **96** and a comparator **97**. Comparator **97** compares the output port's ID to the port ID appearing on the OUTPORT lines of the ARB_A bus and signals state machine **96** when they match. When they match, and when the output port is not busy receiving a transmission from an input port, state machine **96** asserts the IDLE signal line of the ARB_A bus.

Output port switch interface circuit **70** also includes a 4B5B decoder and deserializer circuit **98** which receives, decodes and deserializes transmissions from the input port via switch **20** to produce an output sequence of ATM cells.

On producing each cell of the sequence, circuit **98** pulses a NEXT_CELL signal which shifts the cell into FIFO buffer **72** of FIG. 7. Circuit **98** also sends SOM and EOM signal signals to state machine **96** to indicate when it detects a start and end of message codes in its input data stream. These signals tell state machine **96** whether the output port is busy or idle.

Ethernet Input Port Architecture

FIG. **9** illustrates Ethernet input port PRE(0) of FIG. **3** in more detailed block diagram form. Ethernet input ports PRE(1)–PRE(M) are similar. Ethernet input port PRE(0) includes an Ethernet bus interface circuit **100** for receiving Ethernet transmissions on input Ethernet line RE(0). Bus interface **100** shifts each 4-bit nibble of a transmission into a FIFO buffer **102**. Also, as the source and destination address fields of an incoming Ethernet packet arrive, interface circuit **100** shifts them into a serial-in, parallel-out shift register **104**. As the last nibble of the incoming Ethernet packet is shifted into FIFO buffer **104**, bus interface **100** also shifts an EOM bit into FIFO buffer **102**. At the same time, it sends a TRANSLATE signal to a translation bus interface circuit **106**. Interface circuit **106** responds by reading the source and destination fields out of shift register **104** and sending them to address translation lookup table **44** of FIG. **3** via the TRANS bus. When translation lookup table **44** returns the switch port ID of the output port that is to receive the packet, interface circuit **106** shifts it into a FIFO buffer **108**. When FIFO buffer **108** is not empty it asserts a request signal (REQ) to a switch interface circuit **110** to indicate that the port is storing a complete Ethernet packet.

Arbiter **28** of FIG. **3** polls the input port by placing the input port's ID on INPORT lines of the ARB_E bus and pulsing a POLL line of the ARB_E bus. When the polled input port is busy forwarding data, or when the REQ signal output of FIFO buffer **108** indicates that FIFO buffer **102** is empty, switch interface circuit **110** pulses an NREQ line of the ARB_E bus to tell the arbiter **28** that is not requesting a new connection. Otherwise if the input port is idle and the REQ signal from FIFO buffer **108** is asserted, switch interface circuit **110** responds to a poll by placing the PORT ID output of FIFO buffer **108** on the OUTPUT lines of the ARB_E bus and pulsing the REQ line of the ARB_E bus. If the output port identified by the ID on the OUTPORT lines is idle, arbiter **28** of FIG. **3** connects the input port to the idle output port and pulses a GRANT line of the ARB_E bus. Switch interface circuit **110** then begins shifting nibbles out of FIFO buffer **102, 4B5B** encoding and serializing them, and forwarding them to the output port via switch **20** of FIG. **3**. The EOM bit shifted out of FIFO buffer **102** with the last nibble of the Ethernet packet signals switch interface **110** to stop forwarding data and to shift the port ID out of FIFO buffer **108** and wait for another poll.

Ethernet Output Port Architecture

FIG. **10** illustrates Ethernet output port PTE(0) of FIG. **3** in more detailed block diagram form. Ethernet output ports PTE(1)–PTE(M) are similar. Output port PTE(0) includes a switch interface circuit **112** for receiving serialized, 4B5B encoded packets from an Ethernet input port via crosspoint switch **20** of FIG. **3**. Circuit **112** decodes and deserializes the packet data to produce an output byte sequence that it shifts into a FIFO buffer **114**. An Ethernet interface circuit **116** shifts out and serializes the byte sequence and forwards it outward as an Ethernet transmission on Ethernet bus line TE(0).

Ethernet Arbitration

FIG. **11** illustrates input port switch interface **110** of FIG. **9** and output port switch interface **112** of FIG. **10** in more detailed block diagram form. Arbitrator **28** of FIG. **3** polls an Ethernet input port by placing its ID on the INPORT lines of the ARB_E bus and pulsing the POLL line of that bus. Input port switch interface **110** includes a comparator **118** for signaling a state machine **120** when the input port ID on an INPORT line of the ARB_E bus matches the input port's ID. In such case state machine **120** responds to the POLL by pulsing the NREQ signal line of the ARB_E bus if the port is busy forwarding data to an output port or if the REQ signal from buffer **108** of FIG. **9** is not asserted. Otherwise state machine **120** turns on a tristate buffer **122** to place the output port ID output of FIFO buffer **108** on the OUTPORT lines of the ARB_E bus and then asserts the REQ signal. If the output port having that ID is idle, arbiter **28** of FIG. **3** establishes a connection between the requesting input port and the idle output port and then pulses the GRANT signal line of the ARB_E bus. State machine **120** signals a code generator circuit **124** to send a 4B5B start of message code via a multiplexer **126** signals crosspoint switch **20** to output port interface circuit **112**. State machine **120** then begins shifting 4-bit nibbles of the Ethernet packet out of FIFO buffer **108** of FIG. **9** and clocking them through a **4B5B** encoder and serializer circuit **128**. The encoded and serialized nibbles then pass through multiplexer **126** and crosspoint switch **20** to output port interface circuit **112**. An EOM bit emerging from FIFO buffer **108** with the last nibble of the Ethernet packet signals state machine **120**. State machine **120** then signals code generator **124** to send an end of message code to output port interface circuit **112**. Thereafter state machine **120** considers the input port idle and awaits another poll.

Output port interface circuit **112** includes a 4B5B decoder and deserializer circuit **130** for converting the 4B5B encoded serial data stream from input port **110** into a byte sequence and loading it into FIFO buffer **114** of FIG. **10**. Circuit **130** also transmits SOM and EOM signals to a state machine **132** when it detects the start of message and end of message codes in the data stream. When the output port is not busy forwarding data from an input port, state machine **132** asserts the IDLE signal line of the ARB_E bus when a comparator **134** signals that the ID on the OUTPORT line of the ARB_E bus matches the output port's ID.

Arbiter

FIG. **12** illustrates arbiter **28** of FIG. **3** in more detailed block diagram form. Arbiter **28** includes a state machine **140** which alternates between ATM and Ethernet arbitration modes. In the ATM mode, state machine **140** clocks a counter **142** to increment the port ID on the INPORT lines of the ARB_A bus and clocks a counter **144** that produces the priority level conveyed on the PRIORITY lines of the ARB_A bus. Each time counter **142** is clocked it increments the input port ID until it overflows. At that point it resets its output count to **0** and signals state machine **140**. When state machine **140** resets counter **144**, counter **144** outputs the highest priority code and thereafter decrements that priority code each time that it is clocked thereafter. Upon counting down to 0, counter **144** resets its output to the highest priority code.

State machine **140** initially resets counter **142**. It then pulses the ARB_A bus POLL signal and waits until it detects an REQ or NREQ signal from the polled input port. If an NREQ signal pulse is returned, state machine **140** clocks counter **142** to select a next input port to poll and, if counter **142** overflows, clocks counter **144** to select a next lower priority. State machine **140** then polls the next input port. If the REQ signal is returned in response to an input port poll, but an output port does not assert the IDLE line of the ARB_A bus, state machine **140** polls a next input port. If state machine **140** detects assertion of an REQ signal from a polled input port and detects an IDLE signal from a requested output port, it sets an output signal A/E to switch a multiplexer **146** so that it passes the input and output port ID's on the ARB_A bus INPORT and OUTPUT lines to a decoder **148**. Decoder **148** converts the input and output port ID's to address and data inputs to RAM **42** of FIG. **3** that will establish a connection between the input and output ports through crosspoint switch **20** of FIG. **3**. State machine **140** next pulses a write signal input to RAM **42** of FIG. **3** to write the data into RAM **42**, thereby establishing the connection between the ports. State machine **140** then pulses the GRANT signal time of the ARB-A bus to tell the input port that it may begin transmitting data. State machine **140** thereafter resets counter **144** to produces the highest priority output code and begins the polling process once again.

Thus for each priority level arbiter **28** polls each ATM input port in turn to determine if that input port is ready to forward an ATM transmission to an idle input port. Arbiter works its way down the priority levels until it finds such an input port, establishes the connection, and then starts over again polling the ports at the highest priority level.

In the Ethernet mode, state machine **140** clocks a counter **150** to increment the port ID on the INPORT lines of the ARB_E bus. State machine **140** initially pulses the ARB_E bus POLL signal and waits until it detects a REQ or NREQ signal from the polled input port. If an NREQ signal pulse is returned, state machine **140** clocks counter **150** to select a next input port to poll then polls the next input port. If the polled input port returns the REQ signal, but a requested output port does not assert the IDLE line of the ARB_A bus, state machine **140** polls a next input port. If a polled input port asserts the REQ signal and a requested output port asserts IDLE signal, state machine **140** sets output signal A/E so that switch multiplexer **146** passes the input and output port ID's on the ARB_E bus INPORT and OUTPORT lines to decoder **148** and then pulses the WRITE signal input to RAM **42** of FIG. **3** to write data into RAM **42**. This establishes the connection between the requesting input port and the requested output port. State machine **140** then pulses the GRANT signal to tell the input port that it may begin forwarding data. State machine **140** thereafter polls a next input port.

Reassembly Unit

FIG. **13** illustrates reassembly unit **34** of FIG. **2** in more detailed block diagram form. Reassembly unit **34** includes an input interface **34A** substantially similar to output port switch interface **70** of FIG. **8** for receiving a serialized, 4B5B encoded ATM cell from an ATM input port and converting it to an output sequence of bytes. Input interface **34A** also handles ATM arbitration via the ARB_A bus in a manner similar to output port interface circuit **70**. Reassembly unit **34** also includes an output interface circuit **34B**, similar to Ethernet input port switch interface circuit **110** of FIG. **11**, for receiving a sequence of 4-bit nibbles forming an Ethernet packet and forwarding them to an Ethernet output port in serialized, 4B5B encoded form. Output interface circuit **34B** uses the ARB_E bus to arbitrate for access to Ethernet output ports in the same manner as interface circuit **110** of FIG. **11**.

Input interface **34A** sequentially shifts the payload portion of incoming ATM cells into a FIFO buffer **152** and also shifts in, along with each cell payload, an EOM bit indicating whether the incoming ATM cell is the last cell of ATM transmission. The header portion of the ATM cells is supplied to a header translation table **156** maintained by the host via the SYSTEM bus. Header translation table **156** converts the information in the header to preamble and source and destination address fields (SOURCE,DEST) of an Ethernet packet and also produces output PORTID data referencing the ID of a switch output port to receive the packet. As the cells are loaded into FIFO buffer **152** a check sum circuit **158** computes the ERROR field of the packet. When input interface circuit **34A** pulses its output EOM signal, state machine **154** sends an output REQ signal to output interface circuit **34B**. Thereafter output interface circuit **34B** responds to a poll from arbiter **28** of FIG. **3** by sending the PORTID from header translation table **156** to the arbiter via the ARB_E bus and requesting a connection. When the arbiter grants the request, output interface circuit **34B** begins sending NEXT NIBBLE signals to state machine **154** indicating when it requires a next nibble of the Ethernet packet to be forwarded. State machine **154** controls a multiplexer **160** which selects from among nibbles of the PREAMBLE, SOURCE and DEST field outputs of header translation table **156**, the ERROR field output of check sum circuit **158**, and the payload output of FIFO buffer **152**. State machine **154** also signals FIFO buffer **152** to shift out payload data when needed. In response to a first set of NEXT NIBBLE signals, state machine **154** routes successive nibbles of the PREAMBLE, SOURCE and DEST fields to output interface circuit **34B**. It then begins routing successive nibbles of the payload output of FIFO buffer **152** while successively shifting out cell payloads as necessary. After detecting an EOM bit emerging from FIFO buffer **152**, state machine **154** routes successive nibbles of the ERROR field to output interface circuit **34B**. State machine **154** then sends an EOM signal to output interface circuit **34B** to indicate the end of the packet.

Segmentation Unit

FIG. **14** illustrates segmentation unit **32** of FIG. **2** in more detailed block diagram form. Segmentation unit **32** includes an input interface **32A**, substantially similar to output port switch interface **112** of FIG. **11**, for receiving serialized, 4B5B encoded Ethernet packet from an Ethernet input port and converting it to an output sequence of bytes. Input interface **32A** also handles Ethernet arbitration via the ARB_E bus in a manner similar to output port interface circuit **112** of FIG. **11**. Segmentation unit **32** further includes an output interface circuit **32B** similar to ATM input port switch interface circuit **62** of FIG. **8** for receiving a sequence of 4-bit nibbles forming ATM cells and forwarding them to an ATM output port in serialized, 4B5B encoded form. Output interface circuit **32B** uses the ARB_A bus to arbitrate for access to ATM output ports in the same manner as interface circuit **62** of FIG. **8**.

Input interface circuit **34A** signals a state machine **162** when it outputs a next byte of an incoming Ethernet packet. The bytes forming the PREAMBLE, SOF, TYPE and ERROR fields of the packet are discarded. State machine **162** signals a header translation table **164** to acquire the bytes forming SOURCE and DEST fields, and shifts bytes forming the PAYLOAD field of each cell into a FIFO buffer **166**. After receiving the SOURCE and DEST fields, header translation table **164**, using information provided by the host via the SYSTEM bus, produces the PORTID of the destination ATM output port, the VPI and VCI fields for the ATM cells, and a PRIORITY code for the ATM transmission. Upon receiving an EOM bit output of input interface circuit **32**A, state machine **162** shifts the EOM bit into FIFO buffer **166**. Thereafter, when arbiter **28** of FIG. **3** polls output interface circuit **32**B, interface circuit **32** forwards the priority code conveyed on the ARB_A line to a comparator **168** and asserts a POLL signal. Comparator **168** compares the code to the priority output of header translation table **164** and signals state machine **162** when they match. If state machine **162** has received an EOM bit from input interface **32**A indicating that FIFO buffer **166** stores a complete Ethernet packet, state machine **162** sends an REQ signal to output interface **32**B. Interface **32**B then requests the arbiter to establish a connection to the ATM output port. When the connection is granted, output interface **32** sends a series of NEXT signal pulses to state machine **162**. In response to each NEXT pulse, state machine **162** switches a multiplexer **170** to select a next nibble of a next ATM cell to be forwarded from among the output of FIFO buffer **166**, the VPI and VCI outputs of header translation table **164** and the output of an error code generator circuit **172** which produces the cell's CRC field. State machine **162** shifts data out of FIFO buffer **166** to obtain data for the PAYLOAD fields for the ATM cells. After detecting an EOM bit emerging from FIFO buffer **166**, state machine **162** sends the CRC output of error code generator **172** to output interface **32**B and then forwards an EOM signal to interface **32**B to signal the end of the ATM transmission.

Thus has been described a network switch that can route both ATM and Ethernet packets and which can translate between Ethernet and ATM protocol packets. Since the network switch translates between ATM and Ethernet protocols only when necessary, system throughput is enhanced. Also, since ATM transmissions are buffered at the input ports and fed to the reassembly unit on a per virtual channel basis without interleaving ATM cells, the reassembly unit need not sort out ATM transmissions and need not include extensive buffer memory for storing sorted ATM transmissions prior to reassembly into Ethernet packets.

While the forgoing specification has described preferred embodiment(s) of the present invention, one skilled in the art may make many modifications to the preferred embodiment without departing from the invention in its broader aspects. For example, while network switch is described as handling Ethernet and ATM protocol transmissions, those skilled in the art will understand that the basic architecture of the switch can be adapted to handle routing and conversion of other network protocols. The appended claims therefore are intended to cover all such modifications as fall within the true scope and spirit of the invention.

What is claimed is:

1. A network routing switch for receiving first protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a plurality of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a plurality of second ports, each for forwarding second protocol transmissions to an external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission; and

routing means for selectively interconnecting said first and second ports and said first conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel,

wherein a last cell of each first protocol transmission includes an end of message code identifying said last cell as such,

wherein at least one of said first ports receives cells of separate first protocol transmissions from external network sources in interleaved fashion,

wherein each of said first ports stores cells of received first protocol data transmissions according to the virtual channel referenced by their virtual channel data, and

wherein each of said first ports sends a stored first protocol transmission through said routing means only after receiving and storing said last cell of said first protocol transmission.

2. A network routing switch for receiving first protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a plurality of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a plurality of second ports, each for forwarding second protocol transmissions to an external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission; and

routing means for selectively interconnecting said first and second ports and said first conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel, and

wherein each said first port includes means for determining, from the channel data included in the cells of each first protocol transmission received from an external network device, whether to send said first protocol transmission through said routing means to another of said first ports or to said first conversion means.

3. A network routing switch for receiving first protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a plurality of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a plurality of second ports, each for forwarding second protocol transmissions to an external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission; and

routing means for selectively interconnecting said first and second ports and said first conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein each first protocol transmission includes a priority data value assigning a priority to said first protocol transmission, and

wherein said routing means routes first protocol transmissions to any of said first ports and to said first conversion means with a priority determined by the priority data values included in the first protocol transmissions.

4. The network routing switch in accordance with claim 1 wherein each first protocol transmission includes a priority

data value assigning a priority to said first protocol transmission, and

wherein each of said first ports also stores cells of received first protocol transmissions according to the priority assigned to the transmission.

5. A network routing switch for receiving first protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a plurality of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a plurality of second ports, each for forwarding second protocol transmissions to an external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission; and

routing means for selectively interconnecting said first and second ports and said first conversion means for conveying transmissions therebetween,

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel, and

wherein each second protocol transmission consists of a single packet, said packet including a data payload and a network destination address of a network device to receive the packet.

6. The network routing switch in accordance with claim 5 wherein each second port determines from the network destination address included in each packet received from an external network device whether to send said packet through said routing means to another of said second ports or to said second conversion means.

7. A network routing switch for receiving first protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a plurality of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a plurality of second ports, each for forwarding second protocol transmissions to an external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission; and

routing means for selectively interconnecting said first and second ports and said first conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel,

wherein a last cell of each first protocol transmission includes an end of message code identifying said last cell as such,

wherein at least one of said first ports receives cells of separate first protocol transmission from external network sources in interleaved fashion,

wherein each of said first ports stores cells of received first protocol transmissions according to the virtual channel referenced by their virtual channel data,

wherein each of said first ports sends a stored first protocol transmission through said routing means only after receiving and storing said last cell of said first protocol transmission, and

wherein each said first port includes means for determining, from the channel data included in the cells of each first protocol transmission received from an external network device, whether to send said first protocol transmission through said routing means to another of said first ports or to said first conversion means.

8. The network routing switch in accordance with claim **7** wherein each first protocol transmission includes a priority data value assigning a priority to said first protocol data transmission, and

wherein said routing means routes first protocol transmissions to any of said first ports and to said first conversion means with a priority determined by the priority data values included in the first protocol transmissions.

9. The network routing switch in accordance with claim **8** wherein each of said first ports also stores cells of received first protocol transmissions according to the priority assigned thereto.

10. The network routing switch in accordance with claim **9** wherein each said second protocol transmissions consists of a single packet, said packet including a data payload and a network destination address of a network device to receive the packet.

11. A network routing switch for receiving differing first and second protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, for converting second protocol transmissions into first protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a set of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a set of second ports, each for receiving second protocol transmissions from an external device and for forwarding second protocol transmissions to said external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission;

second conversion means for converting a second protocol transmission into a first protocol transmission; and

routing means for selectively interconnecting said first and second ports and said first and second conversion means for conveying transmissions therebetween,

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein each second port receiving a second protocol transmission from an external network device stores the second protocol transmission until it is complete and then sends the stored second protocol transmission through said routing means to another of said second ports when the second protocol transmission is to be forward to an external network device as a second protocol transmission, and sends the stored second protocol transmission through said routing means to said second conversion means when the stored second protocol transmission is to be converted to a first protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein the second conversion unit converts each second protocol transmission forwarded thereto into a first protocol transmission and sends the first protocol transmission through said routing means to one of said first ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel,

wherein a last cell of each first protocol transmission includes an end of message code identifying said last cell as such, wherein at least one of said first ports receives cells of separate first protocol transmission from external network sources in interleaved fashion,

wherein each of said first ports stores cells of received first protocol transmissions according to the virtual channel referenced by their virtual channel data, and

wherein each of said first ports sends a stored first protocol data transmission through said routing means only after receiving and storing said last cell of said first protocol data transmission.

12. A network routing switch for receiving differing first and second protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, for converting second protocol transmissions into first protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a set of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a set of second ports, each for receiving second protocol transmissions from an external device and for forwarding second protocol transmissions to said external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission; and

second conversion means for converting a second protocol transmission into a first protocol transmission;

routing means for selectively interconnecting said first and second ports and said first and second conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein each second port receiving a second protocol transmission from an external network device stores the second protocol transmission until it is complete and then sends the stored second protocol transmission through said routing means to another of said second ports when the second protocol transmission is to be forward to an external network device as a second protocol transmission, and sends the stored second protocol transmission through said routing means to said second conversion means when the stored second protocol transmission is to be converted to a first protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein the second conversion unit converts each second protocol transmission forwarded thereto into a first protocol transmission and sends the first protocol transmission through said routing means to one of said first ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel, and

wherein each said first port includes means for determining, from the channel data included in the cells of each first protocol transmission received from an external network device, whether to send said first protocol transmission through said routing means to another of said first ports or to said first conversion means.

13. A network routing switch for receiving differing first and second protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, for converting second protocol transmissions into first protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a set of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a set of second ports, each for receiving second protocol transmissions from an external device and for forwarding second protocol transmissions to said external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission;

second conversion means for converting a second protocol transmission into a first protocol transmission; and

routing means for selectively interconnecting said first and second ports and said first and second conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein each second port receiving a second protocol transmission from an external network device stores the second protocol transmission until it is complete and then sends the stored second protocol transmission through said routing means to another of said second ports when the second protocol transmission is to be forward to an external network device as a second protocol transmission, and sends the stored second protocol transmission through said routing means to said second conversion means when the stored second protocol transmission is to be converted to a first protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein the second conversion unit converts each second protocol transmission forwarded thereto into a first protocol transmission and sends the first protocol transmission through said routing means to one of said first ports to be forwarded thereby to an external network device,

wherein each first protocol transmission includes a priority data value assigning a priority to said first protocol transmission, and

wherein said routing means routes first protocol transmissions to any of said first ports and to said first conversion means with a priority determined by the priority data values included in the first protocol transmissions.

14. The network routing switch in accordance with claim 11 wherein each first protocol transmission includes a priority data value assigning a priority to said first protocol transmission, and

wherein each of said first ports also stores cells of received first protocol transmissions according to the priority assigned to the transmission.

15. A network routing switch for receiving differing first and second protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, for converting second protocol transmissions into first protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a set of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a set of second ports, each for receiving second protocol transmissions from an external device and for forwarding second protocol transmissions to said external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission;

second conversion means for converting a second protocol transmission into a first protocol transmission; and

routing means for selectively interconnecting said first and second ports and said first and second conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein each second port receiving a second protocol transmission from an external network device stores the second protocol transmission until it is complete and then sends the stored second protocol transmission through said routing means to another of said second ports when the second protocol transmission is to be

forward to an external network device as a second protocol transmission, and sends the stored second protocol transmission through said routing means to said second conversion means when the stored second protocol transmission is to be converted to a first protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein the second conversion unit converts each second protocol transmission forwarded thereto into a first protocol transmission and sends the first protocol transmission through said routing means to one of said first ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel, and

wherein each said second protocol transmissions consists of a single packet, said packet including a data payload and a network destination address of a network device to receive the packet.

16. The network routing switch in accordance with claim 15 wherein each second port determines from the network destination address included in each packet received from an external network device whether to send said packet through said routing means to another of said second ports or to said second conversion means.

17. A network routing switch for receiving differing first and second protocol transmissions from external network devices, for converting first protocol transmissions to second protocol transmissions, for converting second protocol transmissions into first protocol transmissions, and for forwarding received and converted first and second protocol transmissions to external network devices, the network switch comprising:

a set of first ports, each for receiving first protocol transmissions from an external network device and for forwarding first protocol transmissions to said external network device;

a set of second ports, each for receiving second protocol transmissions from an external device and for forwarding second protocol transmissions to said external network device;

first conversion means for converting a first protocol transmission into a second protocol data transmission;

second conversion means for converting a second protocol transmission into a first protocol transmission; and

routing means for selectively interconnecting said first and second ports and said first and second conversion means for conveying transmissions therebetween;

wherein each first port receiving a first protocol transmission from an external network device stores the first protocol transmission and then sends the stored first protocol transmission through said routing means to another of said first ports when the first protocol transmission is to be forwarded to an external network device as a first protocol transmission, and sends a stored first protocol transmission through said routing means to said first conversion means when the stored first protocol transmission is to be converted to a second protocol transmission before being forwarded to an external network device,

wherein each second port receiving a second protocol transmission from an external network device stores the second protocol transmission until it is complete and then sends the stored second protocol transmission through said routing means to another of said second ports when the second protocol transmission is to be forward to an external network device as a second protocol transmission, and sends the stored second protocol transmission through said routing means to said second conversion means when the stored second protocol transmission is to be converted to a first protocol transmission before being forwarded to an external network device,

wherein the first conversion unit converts each first protocol transmission forwarded thereto into a second protocol transmission and sends the second protocol transmission through said routing means to one of said second ports to be forwarded thereby to an external network device,

wherein the second conversion unit converts each second protocol transmission forwarded thereto into a first protocol transmission and sends the first protocol transmission through said routing means to one of said first ports to be forwarded thereby to an external network device,

wherein each said first protocol transmission received by a first port comprises a plurality of sequentially transmitted cells, each cell including channel data referencing a virtual channel,

wherein a last cell of each first protocol transmission includes an end of message code identifying said last cell as such, wherein at least one of said first ports receives cells of separate first protocol transmission from external network sources in interleaved fashion,

wherein each of said first ports stores cells of received first protocol transmissions according to the virtual channel referenced by their virtual channel data,

wherein each of said first ports send a stored first protocol transmission through said routing means only after receiving and storing said last cell of said first protocol transmission, and

wherein each said first port includes means for determining, from the channel data included in the cells of each first protocol transmission received from an external network device, whether to send said first protocol transmission through said routing means to another of d first ports or to said first conversion means.

18. The network routing switch in accordance with claim **17** wherein each first protocol transmission includes a priority data value assigning a priority to said first protocol transmission, and

wherein said routing means routes first protocol transmissions to said first ports and said first conversion means in order of priority indicated by the priority data values included in the first protocol transmissions.

19. The network routing switch in accordance with claim **18** wherein each of said first ports also stores cells of received first protocol transmissions in groups linked according to the priority assigned to the transmission.

20. The network routing switch in accordance with claim **19** wherein each said second protocol transmission consists of a single packet, said packet including a data payload and a network destination address of a network device to receive the packet.

21. The network routing switch in accordance with claim **20** wherein each second port determines from the network destination address included in each packet received from an external network device whether to route said packet through said routing means to another of said second ports or to said second conversion means.

* * * * *

US006570876B1

(12) **United States Patent**
Aimoto

(10) **Patent No.: US 6,570,876 B1**
(45) **Date of Patent: May 27, 2003**

(54) PACKET SWITCH AND SWITCHING METHOD FOR SWITCHING VARIABLE LENGTH PACKETS

(75) Inventor: **Takeshi Aimoto**, Sagamihara (JP)

(73) Assignee: **Hitachi, Ltd.**, Tokyo (JP)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/280,980**

(22) Filed: **Mar. 30, 1999**

(30) **Foreign Application Priority Data**

Apr. 1, 1998 (JP) 10-088330

(51) **Int. Cl.**^{7} **H04L 12/56**; H04L 12/54

(52) **U.S. Cl.** **370/389**; 370/412; 370/413; 370/415; 370/417; 370/422; 370/428; 370/468; 370/474

(58) **Field of Search** 370/389, 391, 370/412, 413, 415, 417, 422, 428, 465, 468, 474, 912, 461, 462; 709/223, 226, 232

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,231,633 A * 7/1993 Hluchyj et al. 370/94.1
5,268,900 A * 12/1993 Hluchyj et al. 370/94.1
6,067,301 A * 5/2000 Aatresh 370/418
6,104,700 A * 8/2000 Haddock et al. 370/235

FOREIGN PATENT DOCUMENTS

JP 7135512 5/1995

* cited by examiner

Primary Examiner—Hassan Kizou
Assistant Examiner—Joe Logsdon
(74) *Attorney, Agent, or Firm*—Antonelli, Terry, Stout & Kraus, LLP

(57) **ABSTRACT**

A packet switch for switching variable length packets, wherein each of output port interfaces includes a buffer memory for storing transmission packets, a transmission priority controller for classifying, based on a predetermined algorithm, transmission packets passed from a packet switching unit into a plurality of queue groups to which individual bandwidths are assigned respectively, and queuing said transmission packets in said buffer memory so as to form a plurality of queues according to transmission priority in each of said queue groups, and a packet read-out controller for reading out said transmission packets from each of said queue groups in the buffer memory according to the order of transmission priority of the packets while guaranteeing the bandwidth assigned to the queue group.

6 Claims, 7 Drawing Sheets

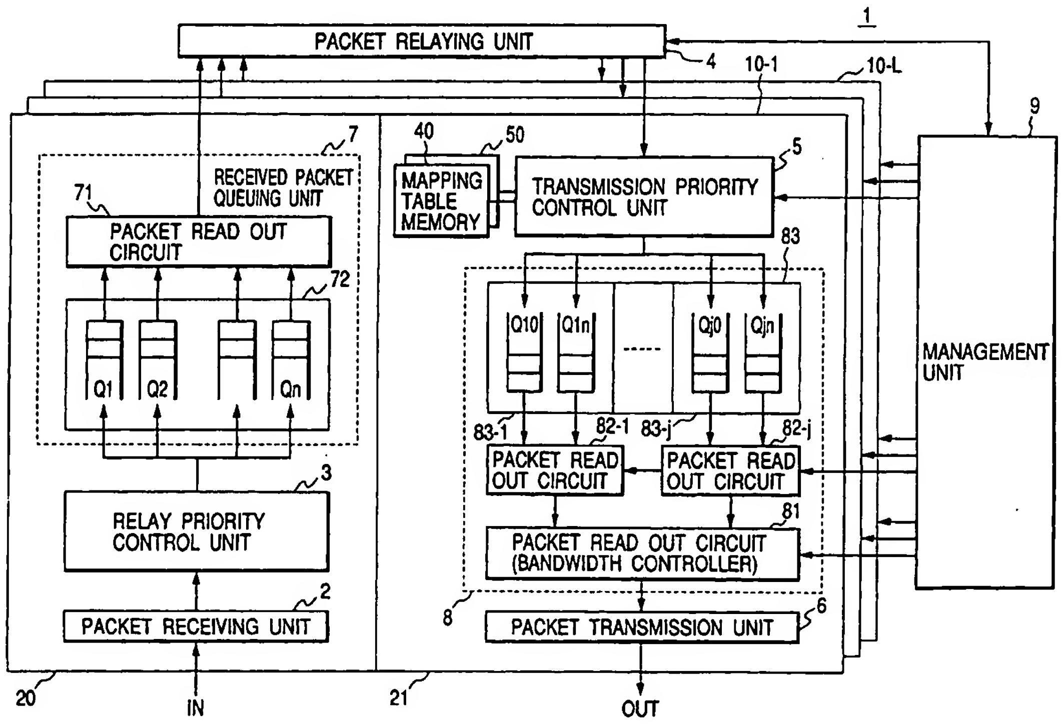
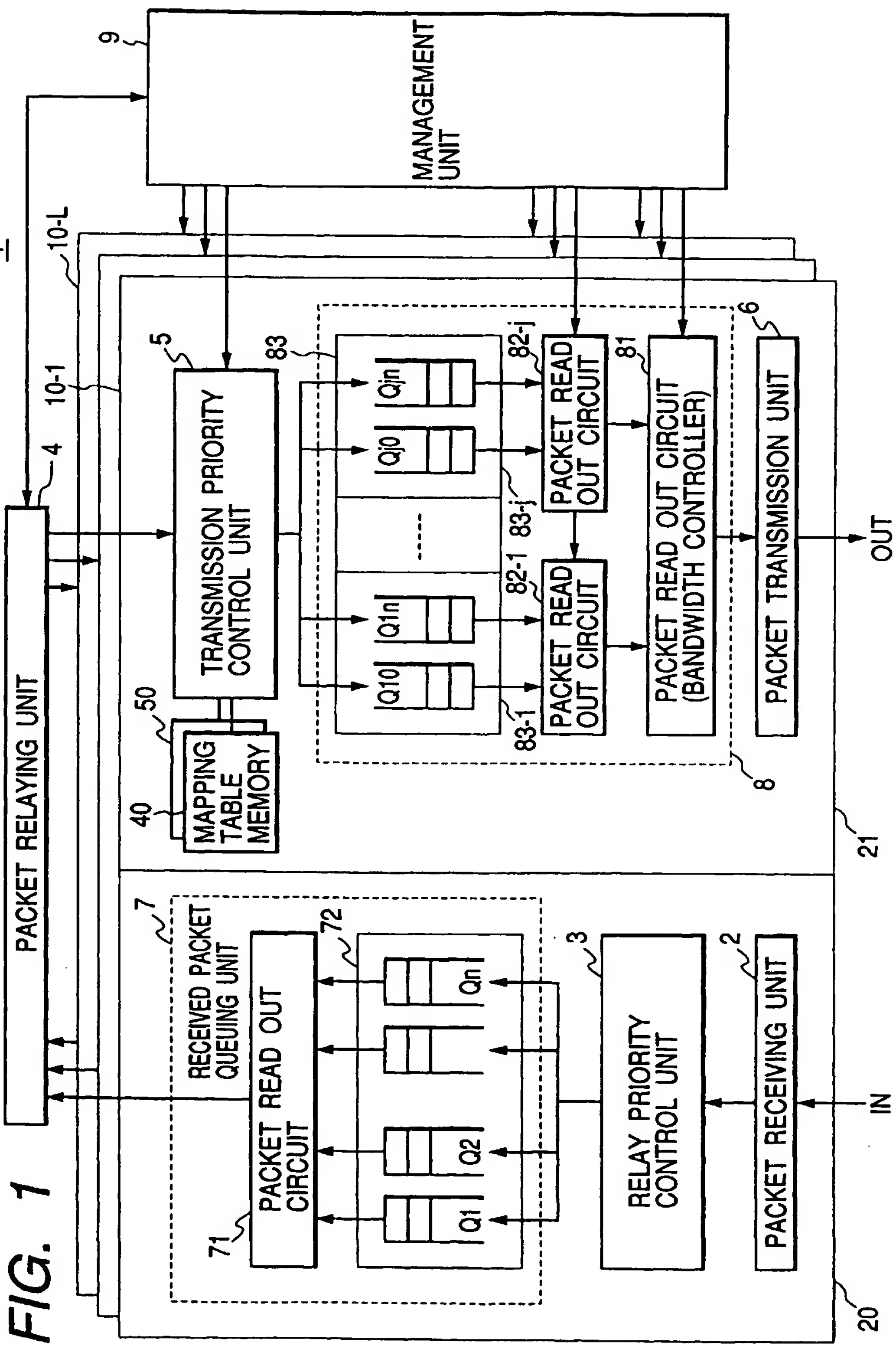


FIG. 1



1
9
MANAGEMENT UNIT
10-L
10-1
4
PACKET RELAYING UNIT
5
TRANSMISSION PRIORITY CONTROL UNIT
50
40
MAPPING TABLE MEMORY
83
Qjn
Qj0

Q1n
Q10
83-j
83-1
82-j
82-1
PACKET READ OUT CIRCUIT
PACKET READ OUT CIRCUIT
81
PACKET READ OUT CIRCUIT (BANDWIDTH CONTROLLER)
6
PACKET TRANSMISSION UNIT
OUT
8
21
7
RECEIVED PACKET QUEUING UNIT
71
PACKET READ OUT CIRCUIT
72
Qn
Q2
Q1
3
RELAY PRIORITY CONTROL UNIT
2
PACKET RECEIVING UNIT
IN
20

FIG. 2

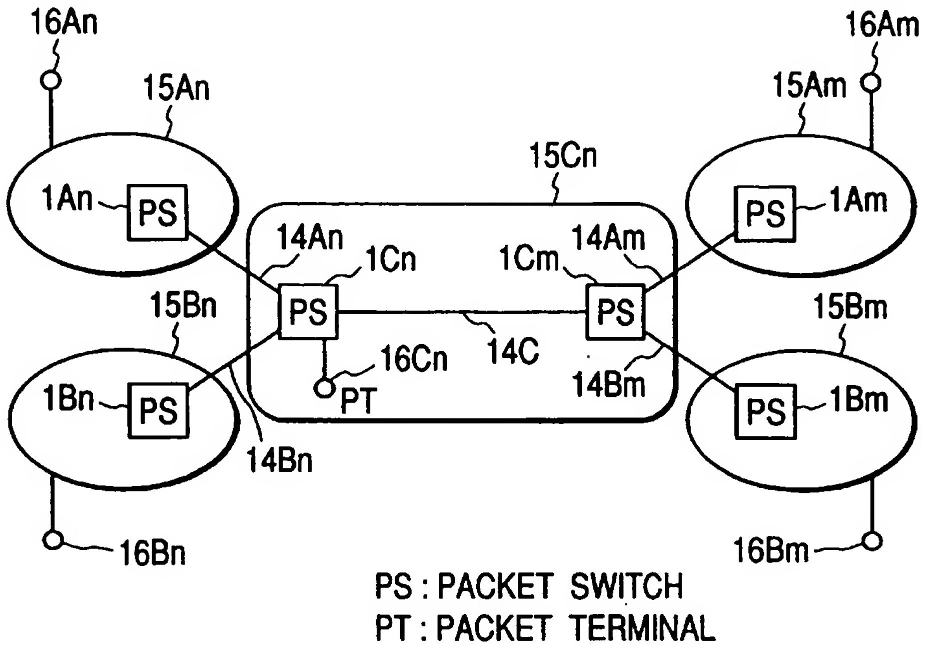

16An
15An
1An
PS
16Am
15Am
1Am
PS
15Cn
14An
1Cn
1Cm
14Am
PS
PS
15Bn
15Bm
16Cn
14C
14Bm
PT
1Bn
PS
PS
1Bm
14Bn
16Bn
16Bm
PS : PACKET SWITCH
PT : PACKET TERMINAL

FIG. 3

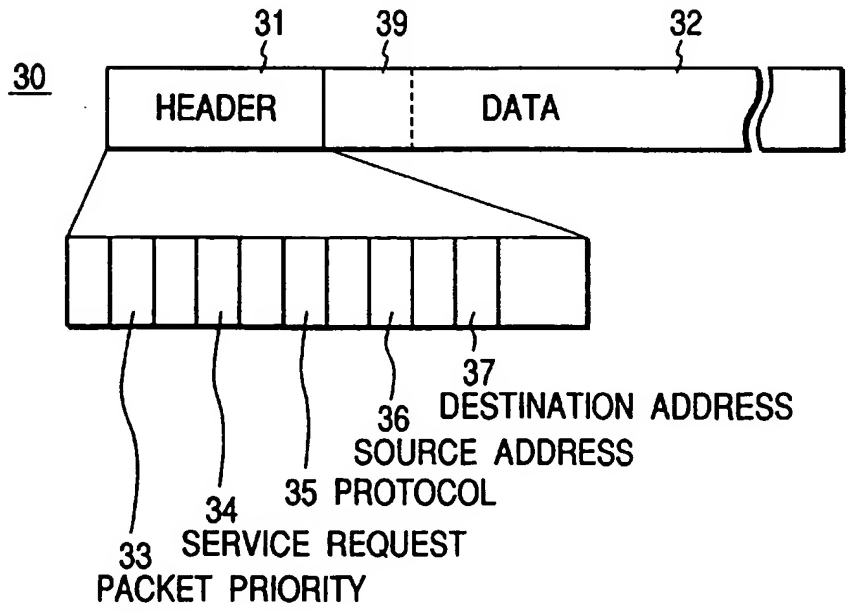

31
39
32
30
HEADER
DATA
37
DESTINATION ADDRESS
36
SOURCE ADDRESS
35 PROTOCOL
34
SERVICE REQUEST
33
PACKET PRIORITY

FIG. 4

40

41 42

PROTOCOL / PRIORITY	A	B	C	----------------	ETC.
0 (MIN)	4	0	4		0
1	4	1	4		1
2	4	2	4		2
3	4	3	4		3
4	4	3	4		4
5	4	3	5		5
6	4	3	6		6
7 (MAX)	4	3	7		7

FIG. 5

50

51 52

SOURCE / DESTINATON	An	Bn	Cn	----------------	ETC.
Am	1 (BW1)	—	—		—
Bm	—	2 (BW2)	—		—
Cm	—	—	3 (BW3)		—
Dm	—	—	—	4 (BW4)	—
Em	—	—	—		—
	—	—	—		—
	—	—	—		—
ETC.	—	—	—		j (BWj)

— : PACKET DISCARD

RESIDUAL BANDWIDTH

FIG. 6

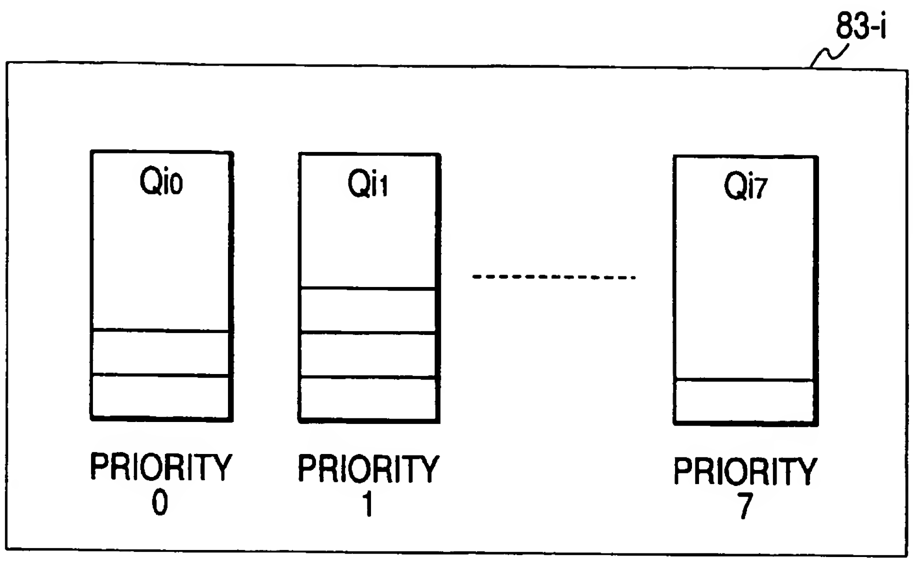
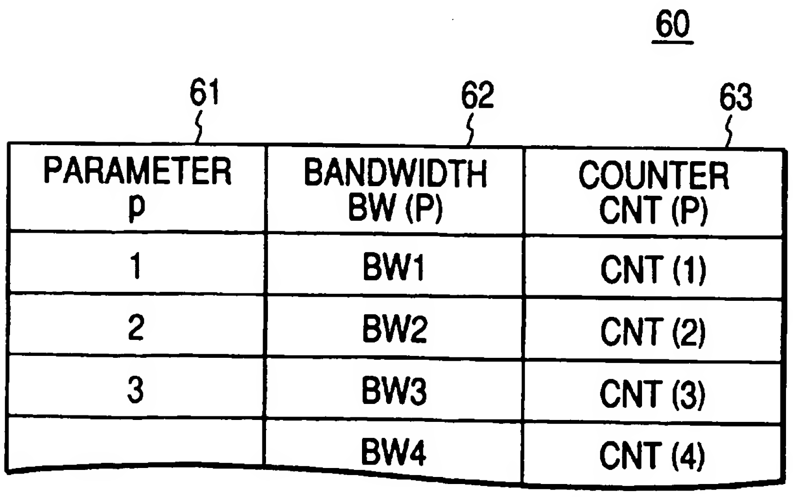
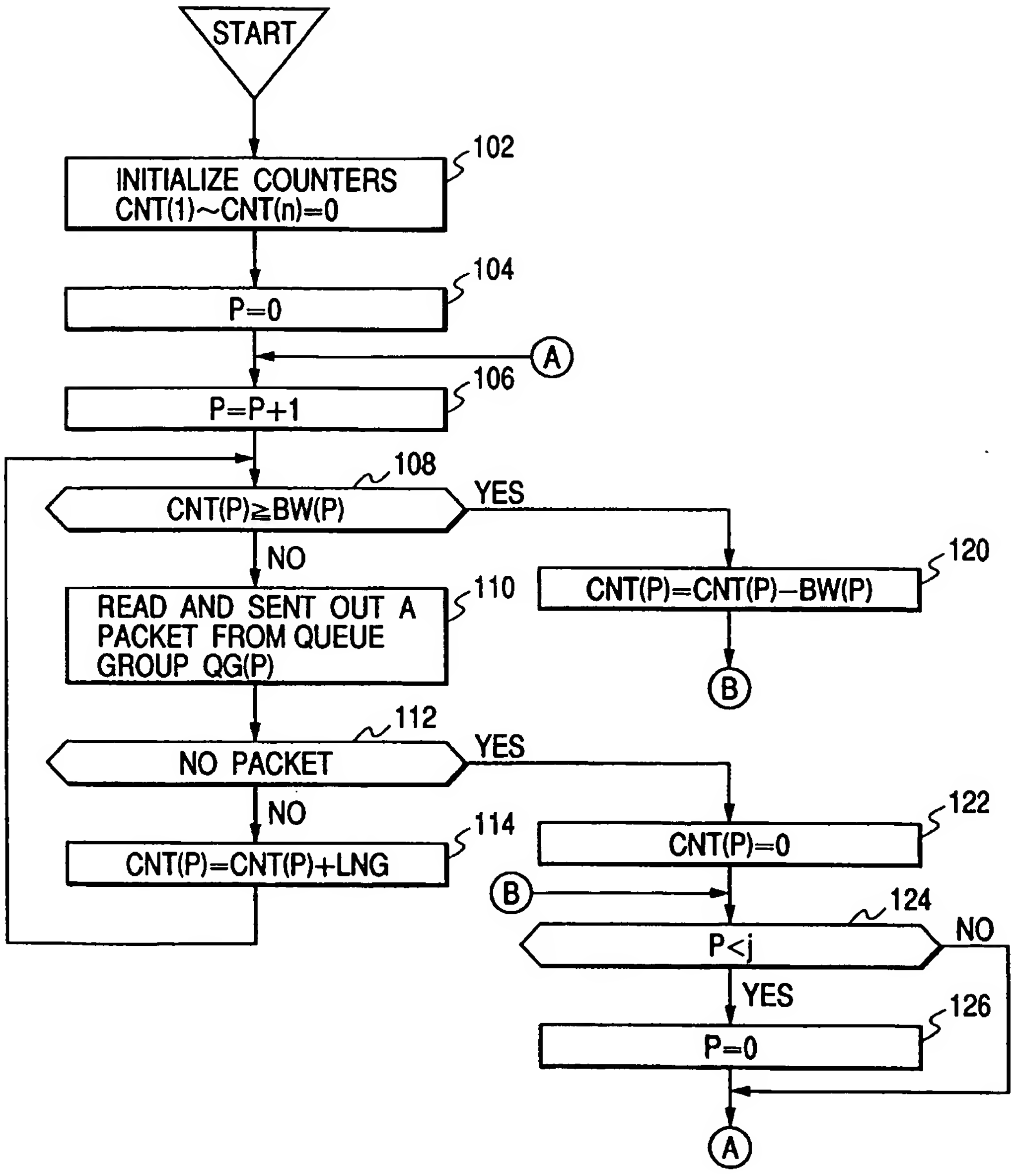


FIG. 7



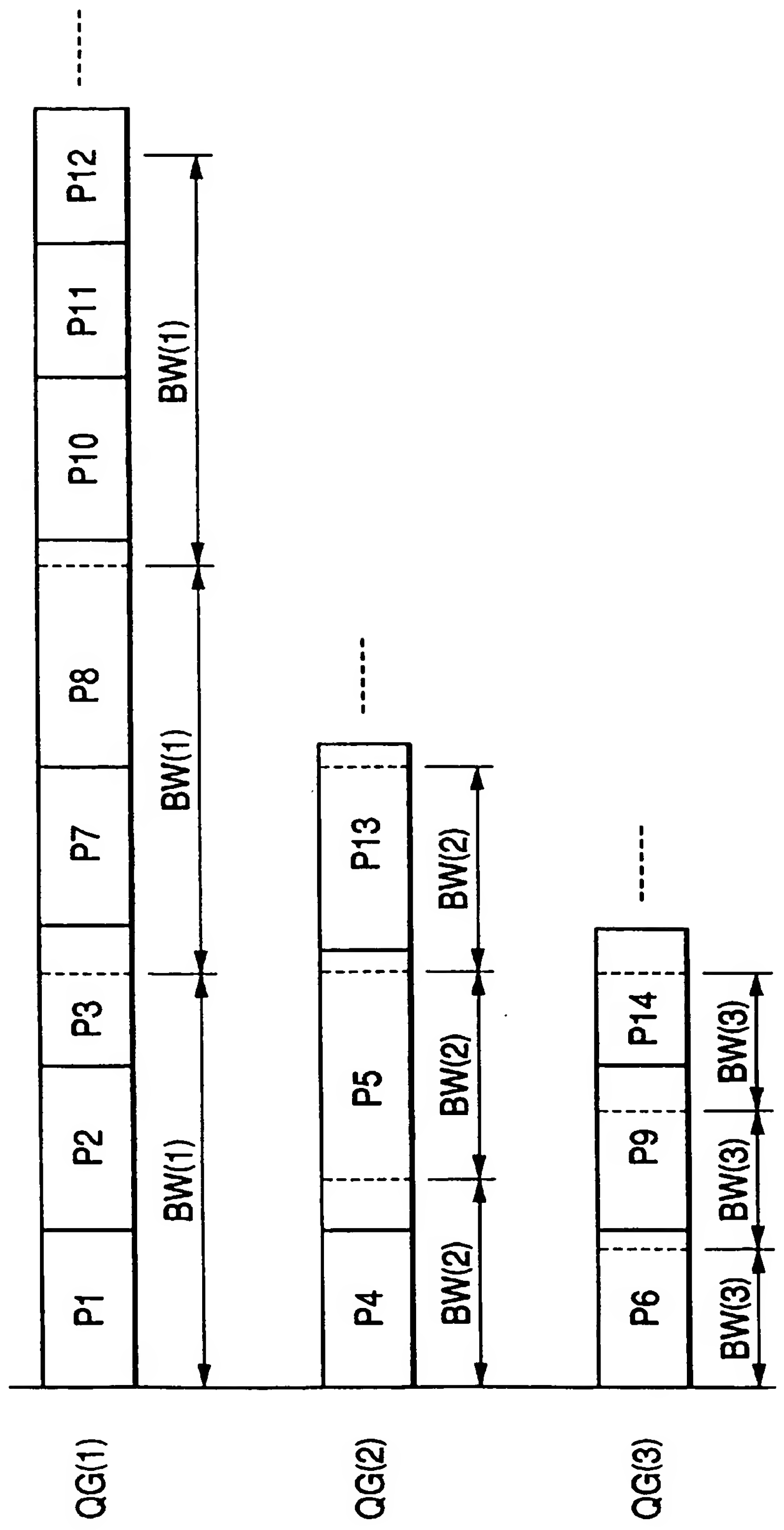
PARAMETER p	BANDWIDTH BW (P)	COUNTER CNT (P)
1	BW1	CNT (1)
2	BW2	CNT (2)
3	BW3	CNT (3)
	BW4	CNT (4)

FIG. 8



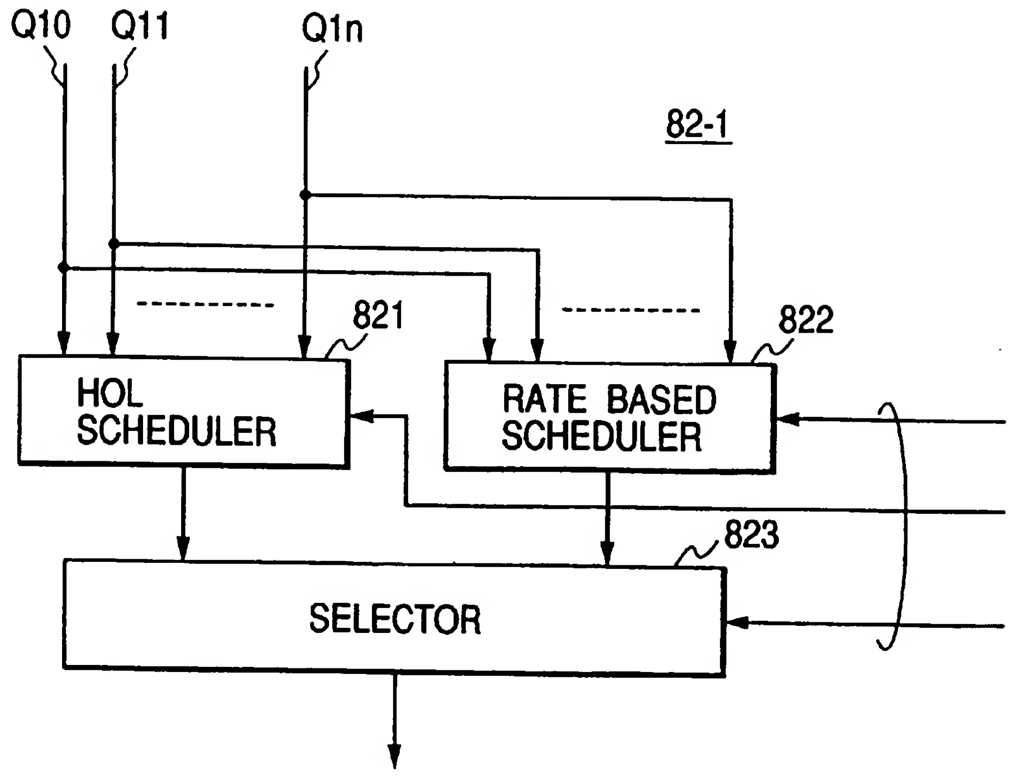
START
INITIALIZE COUNTERS CNT(1)~CNT(n)=0
102
P=0
104
A
P=P+1
106
CNT(P)≧BW(P)
108
YES
NO
CNT(P)=CNT(P)−BW(P)
120
B
READ AND SENT OUT A PACKET FROM QUEUE GROUP QG(P)
110
NO PACKET
112
YES
NO
CNT(P)=0
122
CNT(P)=CNT(P)+LNG
114
B
P<j
124
NO
YES
P=0
126
A

FIG. 9



QG(1)
QG(2)
QG(3)
P1
P2
P3
P7
P8
P10
P11
P12
P4
P5
P13
P6
P9
P14
BW(1)
BW(2)
BW(3)

FIG. 10


Q10
Q11
Q1n
82-1
821
822
823
HOL SCHEDULER
RATE BASED SCHEDULER
SELECTOR

PACKET SWITCH AND SWITCHING METHOD FOR SWITCHING VARIABLE LENGTH PACKETS

BACKGROUND OF THE INVENTION

(1) Field of the Invention

The present invention relates to a packet switch and a switching method, more particularly to a packet switch and a switching method for switching variable length packets by controlling transmission of packets according to the priority of respective packets. More concretely, the present invention relates to a packet switch and a switching method suitable for switching variable length packets in the Internet, as well as intranets, and capable for transmitting the variable length packets by controlling a bandwidth thereof.

(2) Description of the Related Art

The packet communication is a communication method for transmitting and receiving data in a form of a packet comprising a predetermined format of header and a data block obtained by dividing transmission data into blocks each having a proper length, for example, 48 byte to 1.5 Kbytes in length. A packet switch adopts a so-called stored and forward switching method, in which switching of communication data is carried out after the communication data as stored temporarily in the switch, so that it is able to apply a variety of control to the packets passing through the switch.

In the case of the Internet coming into wide use rapidly in recent years, data is communicated as variable length packets with an IP (Internet Protocol) header respectively. Therefore, a packet switching technique is essential to each node for connecting a network to another. An inter-networks connecting apparatus (node apparatus) such as a router is also provided with the packet switching function even if it is not called a "packet switch" actually. Consequently, in the present invention, every network apparatus provided with the packet switching function, including the node apparatus having such a specific name as router, will be referred to as a packet switch.

As a prior art related to such the packet switching, a router is disclosed in Japanese Unexamined Patent Publication No.7-135512. The prior art intends to provide a router with a function for controlling the transfer of received packets according to the priority thereof and a function for controlling the discard of packets performed when a buffer congestion occurs. According to the prior art, packets received by a packet receiving unit are queued in a buffer memory by a priority control unit according to the transmission priority and the discard priority of each packet, then they are transmitted to a packet transmission unit according to the order of the transmission priority. The priority control unit refers to a mapping table based on the transmission priority information and the protocol information included in the header of each received packet to obtain the processing priority and the discard priority corresponding to the received packet, and stores the packet in one of queues prepared corresponding to the discard priority for each processing priority. Stored packets are output sequentially according to the order of descending processing priority. When the free area capacity of the buffer memory goes under a predetermined threshold value, discard control of the stored packets is carried out to avoid congestion. In this discard control, packets with higher discard priority are discarded from queues according to the order of ascending processing priority until the free area capacity of the buffer memory reaches a target threshold value.

In the case of the above related art, only one type of the mapping table is referred for queuing packets. The transmission priority and the discard priority of each packet are defined in the mapping table corresponding to the priority information and the protocol information included in the header of each of received packets. Consequently, in the case of the related art, it is difficult to control a bandwidth for each packet flow and to provide peculiar packet switching services depending on, for example, the source or destination network of those packets in a node apparatus connecting a plurality of networks such as the Internet.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a packet switch and a packet switching method that can control a bandwidth of each of variable length packet flows.

It is another object of the present invention to provide a packet switch and a packet switching method that can guarantee the bandwidth previously reserved for each of variable length packet flows by a communication service contract.

It is still another object of the present invention to provide a packet switch and a packet switching method that can perform both bandwidth control and priority control for each variable length packet flow.

It is further another object of the present invention to provide a packet switch and a packet switching method that can perform both bandwidth control and priority control according to the communication protocol of variable length packets.

In order to achieve the above objects, the packet switch of the present invention comprises a plurality of input port interfaces, a plurality of output port interfaces, and a packet switching unit for switching each of variable length packets received from the input port interfaces to one of the output port interfaces corresponding to the destination address of the packet, wherein each of the output port interfaces comprises a buffer memory for storing transmission packets, a transmission priority controller configured so as to classify the transmission packets received from the packet switching unit based on a predetermined algorithm and queue each of those transmission packets according to transmission priority thereof into one of a plurality of queue groups, each of which is assigned an individual bandwidth, a transmission packet read-out controller for accessing the queue groups of the buffer memory cyclically to read out transmission packets from each of those queue groups according to the order of descending transmission priority while guaranteeing the bandwidth assigned to each queue group, and a packet transmission circuit for transmitting the transmission packets read out by the transmission packet read-out controller to an output port associated with the output port interface.

According to an embodiment of the present invention, the transmission priority controller is, for example, provided with means for identifying communication service contract and transmission priority related to each of transmission packets according to the header information of the transmission packet received from the packet switching unit, thereby to queue the transmission packet in a queue corresponding to the identified transmission priority in a queue group corresponding to the identified communication service contract.

More concretely, the transmission priority controller includes, for example, a first management table for defining the identifier of the queue group corresponding to the combination of a source network address and a destination

network address and a second management table for defining a transmission priority corresponding to the combination of a communication protocol and a packet priority, thereby to identify a queue group corresponding to each of the transmission packets by referring to the first management table based on the source network address and the destination network address included in the header of the transmission packet, and to identify the transmission priority of the transmission packet by referring to the second management table based on the communication protocol information and the packet priority information included in the header of the transmission packet.

According to another embodiment of the present invention, the transmission priority controller identifies a queue group corresponding to each of the transmission packets based on the protocol information included in the header of the transmission packet, then queues the transmission packet in one of the queues in the identified queue group. In this case, for example, the object of switching is an IP packet which is identified by the protocol of the network layer of the OSI (Open Systems Interconnection) reference model, and transmission packets are classified into a plurality of queue groups according to the protocol type of the transport layer of the OSI reference model, for example, TCP (Transmission Control Protocol), UDP (User Datagram Protocol), ICMP (Internet Control Message Protocol), or IGMP (Internet Group Management Protocol).

In order to guarantee the bandwidth of each queue group, the transmission packet read-out controller of the present invention specifies, for example, a queue group among the queue groups cyclically to read out transmission packets, assigns a packet read duration to the specified queue group depending on the bandwidth thereof, and reads out transmission packets according to the order of descending transmission priority.

Furthermore, according to an embodiment of the present invention, the transmission packet read-out controller includes a management table for defining a threshold value in proportion to the bandwidth corresponding to each of the queue groups and a control means for accessing the plurality of queue groups cyclically to read out transmission packets continuously from each of the queue groups, until the total length of the read out packets exceeds the threshold value defined in the management table and for storing the surplus length of the read out packet exceeding the threshold value as an initial value for use in the counting of the total packet length in the next read cycle.

The controlling means, for example, checks the initial value for counting the total packet length associated with one of the queue groups before reading out transmission packets from the queue group. If the initial value exceeds its threshold value, the controlling means subtracts the threshold value from the initial value and switches the object of packet read out to a next queue group without reading out any packets from the current queue group. Consequently, variable length packets are read out to an output port, while the bandwidth is controlled properly in a long-ranged view.

The algorithm provided for the transmission priority controller to classify transmission packets, as well as the bandwidth control function provided for the transmission packet read-out controller can be modified suitably, for example, in response to a control command issued from an external network management terminal.

Another feature of the packet switch of the present invention resides in that each of the input port interfaces comprises a receiving buffer memory for storing received

packets temporarily, a relaying priority controller configured so as to filter the packets received from input ports based on the destination address of the received packets, classify the filtered received packets according to their priority based on a predetermined algorithm, and queue those classified packets in the receiving buffer memory according to the priority, and a received packet read-out circuit for reading out the received packets from the receiving buffer memory according to the order of priority to supply those packets to the packet switching unit. By transferring the packets according to the order of priority in both of the input port interfaces and the output port interfaces, it is possible to reduce the delay time of high priority packets within the packet switch more significantly.

The switching method of the present invention for switching variable length packets comprises the steps of: (a) filtering received packets received from an input port based on the destination address of each of those packets; (b) transferring the filtered packets to one of the output ports interfaces as transmission packets according to the destination address included in the header of each of those packets; (c) classifying the transmission packets based on a predetermined algorithm and queuing the classified packets according to transmission priority into one of the a plurality of queue groups assigned with a individual bandwidth respectively; (d) accessing the plural queue groups cyclically to read out transmission packets from each of those queues according to the order of transmission priority while guaranteeing the bandwidth assigned to each of the queue groups; and (e) transmitting the read out transmission packets to an output port.

More specifically, the step (b) of transferring received packets preferably includes a step (b1) of queuing received packets according to priority in the receiving buffer and a step (b2) of reading out the received packets stored in the receiving buffer according to the order of priority and transferring those packets as transmission packets to one of the output interfaces.

In the step (c) for queuing transmission packets, for example, transmission packets are classified by communication service contract related to each of those transmission packets and queued in a queue group corresponding to the communication service contract by transmission priority. Instead of this, transmission packets may be classified by communication protocol related to each of those transmission packets, so that those transmission packets are queued according to transmission priority in a queue group corresponding to the protocol thereof in the step (c).

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram indicating an embodiment of the packet switch of the present invention.

FIG. 2 is a configuration of a network to which the packet switch of the present invention applies.

FIG. 3 is a format of packets to be processed by the packet switch of the present invention.

FIG. 4 is an example of a mapping table **40** for defining a transmission priority corresponding to both protocol and priority of each packet.

FIG. 5 is an example of a mapping table **50** for defining a bandwidth according to both source and destination addresses of each packet.

FIG. 6 is a drawing for explaining the layout of queues formed according to transmission priority in a buffer area **83**-i prepared for each bandwidth.

FIG. 7 is an example of a management table provided for a packet read-out circuit **81** shown in FIG. **1**.

FIG. **8** is a flow chart showing the functions of the packet read-out circuit **81**.

FIG. **9** is a drawing for explaining the order of packets read out by the packet read-out circuit **81**.

FIG. **10** is a block diagram showing another embodiment of the packet read-out circuit **82-1** shown in FIG. **1**.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

At first, the configuration and operation of the packet switch of the present invention will be described briefly with reference to FIG. **1**.

The packet switch I comprises a plurality of control boards (**10-1** to **10-L**) prepared for each of input/output ports, a packet relaying unit (packet switching unit) for relaying packets between those control boards **10**, and a management unit **9** connected to each of the control boards **10**. Each of the control boards **10** comprises an input port interface **20** and an output port interface **21**.

The input port interface **20** includes a packet receiving unit **2** for receiving packets flowing on a network from an input port IN, a relaying priority control unit **3** for storing the packets received from the packet receiving unit in a receiving buffer memory **72** according to the priority, and a received packet queuing unit **7** for transferring the packets stored in the receiving buffer memory **72** to the packet relaying unit **4** according to the priority.

The output port interface **21** includes a transmission priority control unit **5** for storing the packets received from the relaying unit **4** selectively in one of a plurality of queue groups formed in a transmission buffer **83** (**83-1** to **83-**j), a transmission packet queuing unit **8** for reading out the packets stored in the transmission buffer **83** according to the transmission priority while guaranteeing the bandwidth assigned to each queue group, and a packet transmission unit **6** for transmitting the transmission packets read out by the transmission packet queuing unit **8** to an output port OUT connected to a network.

The relaying priority control unit **3** is provided with a filtering function of referencing to a routing table based on the destination address included in the header of each received packet to determine whether the transmission packet should be relayed to another network or not. If it is determined that the transmission packet should be relayed to another network, the packet is given an output port identifier, then queued in the receiving buffer memory **72** according to the priority. More concretely, the relaying priority control unit **3** determines a relaying priority based on the management information (packet priority information, protocol information, network address, etc., which will be described later in detail with reference to FIG. **3**) and stores the received packet in a queue Qi corresponding to the determined relaying priority. The queue Qi is one of a plurality of packet queues Q1 to Qn formed by relaying priority in the receiving buffer memory **72**.

In the received packet queuing unit **7**, packets are read out by the packet read-out circuit **71** sequentially from the queues formed in the receiving buffer memory **72** according to the order of descending priority, then output to the packet relaying unit **4**. The relaying unit **4** comprises, for example, a bus or crossbar switch for switching the packets received from the receiving packet queuing unit **7** provided in each control board **10** to one of the other control boards identified by the output port identifier added to the packet.

The transmission priority control unit **5** stores packets received from the relaying unit **4** in a transmission buffer memory **83** (**83-1** to **83-**j) as transmission packets.

A feature of the present invention resides in that the transmission priority control unit **5** forms a multi-dimensional queue structure in the transmission buffer **83** by classifying transmission packets into a plurality of groups base on the header information of each of those packets and queuing those transmission packets according to the transmission priority for each group.

In the illustrated embodiment, the transmission buffer **83** is divided into a plurality of buffer areas, each of which includes a plurality of queues (Q10 to Q1n) . . . (Qj0 to Qjn), so that a total of j queue groups **83-1** to **83-**j are formed. The transmission priority control unit **5**, as will be described later, refers to the mapping tables **40** and **50**, which will be described later with reference to FIGS. **4** and **5**, based on the control information (e.g., packet priority information, protocol information, network address, etc. as will be described later in detail with reference to FIG. **3**) included in the header of each packet, thereby determining the transmission priority and the queue group of the transmission packet. Each of the queue groups, as will be described later, is assigned with an individual bandwidth, and each of transmission packets is stored in the queue Qik in one of those queue groups corresponding to the transmission priority.

The transmission packet queuing unit **8** includes first stage read-out circuits **82** (**82-1** to **82-**j) for reading out packets according to the transmission priority from each queue group (buffer area **83-**i) and a next stage read-out circuit **81** for outputting the packets read out by those first-stage read-out circuits to the transmission unit **6** according to the bandwidth assigned to each queue group.

The packet transmission unit **6** transmits packets received from the read-out circuit **81** to an output port OUT. The output port identifier added by the relaying priority control unit **3** to each received packet is removed by the transmission priority control unit **5** or the packet transmission unit **6**.

Grouping of transmission packets performed by the transmission priority control unit **5** is made, to put in the concrete, for each communication service contract between a line provider and a network user or customer user group. The read-out circuit **81** reads out transmission packets sequentially from each of the queue groups so as to adapt to the communication service level promised by each contract.

A communication service contract mentioned here means a bandwidth guarantee contract exchanged between a network user and a network line provider (or an Internet communication provider). More concretely, it means a contract between a business company or a public agency and a line provider. According to such a contract, a communication service level including such items as communication quality (e.g., packet delay time, delay variability, packet discard rate, etc.), an assigned bandwidth, etc. is guaranteed for each user. Each of those network line providers is thus responsible for the contracted communication service level with respect to each packet flow on the subject network.

A network manager can change the packet queuing algorithm in the transmission priority control unit **5** and the read-out algorithm in the read-out circuits **81** and **82** via the management unit **9** by issuing a command, for example, from the management terminal PT shown with a reference numeral **16**Cn in FIG. **2**.

The packet queuing algorithm used by the transmission priority control unit **5** can be changed by updating the contents in the mapping tables **40** and **50** and the packet read-out algorithm used by the read-out circuit **81** can be changed by updating the contents in the management table **60** as will be described later.

Next, a configuration of the packet switch of the present invention, as well as how to use the packet switch will be described with reference to FIG. **2**.

The network shown in FIG. **2** comprises sub-networks **15An**, **15Bn**, **15Am**, and **15Bm**, and a backbone network **15Cn** for connecting those sub-networks to each other. Each of the sub-networks **15An** to **15Bm** is connected to packet communication terminals **16An**, **16Bn**, **16Am**, and **16Bm** respectively.

Each of the sub-networks **15An** to **15Bm** is provided with packet switches **1An**, **1Bn**, **1Am**, and **1Bm**. The packet switches **1An** and **1Bn** are connected to a packet switches **1Cn** in the network **15Cn** through the lines **14An** and **14Bn** respectively. The packet switches **1Am** and **1Bm** are connected to a packet switch **1Cm** in the network **15Cn** through the lines **14Am** and **14Bm** respectively.

In the backbone network **15Cn** are connected the packet switches **1Cn** and **1Cm** to each other through a line **14C**.

The packet switches **1** of the present invention shown in FIG. **1** are used as packet switches **1Cn** and **1Cm** of the backbone network. **16Cn** indicates a management terminal operated by the manager of the network **15Cn**. According to the commands issued from this management terminal **16Cn**, the control information held in the packet switches **1Cn** and **1Cm**, that is, the contents in the tables **40**, **50**, and **60** can be updated.

Generally, a packet communication can perform a multiplex communication, wherein a bandwidth indicating a communication capability of a line per unit time is divided to assign the bandwidth to a plurality of users.

In the case of the network shown in FIG. **2**, the communication line **14C** of the backbone network is shared by the communication between the sub-networks **15An** and **15Am** and the communication between the sub-networks **15Bn** and **15Bm**. Packet communication allows each line to be used effectively through the statistical multiplexing effect by sharing a line among a plurality of communications such way and by assigning a bandwidth to each of those communications only when it is needed actually.

In this case, it is assumed that the communication between the sub-networks **15An** and **15Am** is executed under the contract A and the communication between the sub-networks **15Bn** and **15Bm** is executed under the contract B, and the network provider who owns the backbone network **15C** provides packet flows on the line **14C** with packet switching services according to the service levels guaranteed by the contracts A and B. This example is equivalent to a case in which the sub-networks **15An** and **15Am** are, for example, of the head office and a branch office of a company A and the sub-network **15Bn** and **15Bm** are of the head office and a branch office of a company B, and each of those companies A and B perform their intranet communications according to the contracts with the Internet communication provider C.

The packet switch of the present invention provides the following two forms of usage for a plurality of contract parties who share the line **14C**.

The first form of usage distributes the bandwidth of the line **14C** to both contracts A and B equally or with a weight according to the contract contents respectively, through the rate based scheduling. If there are a plurality of packet flows for the same contract, packet switching is carried out equally for each of those packet flows or according to the protocol and the packet priority of each packet flow.

The second form of usage allows only one contract, for example, the packet flow of the contract A to use the full bandwidth of the line **14C** through the head-of-line scheduling. In this case, the guaranteed bandwidth is set as 0 for all the packet flows of other contracts except for the contract A, and packet switching for these contracts are carried out using the residual bandwidth of the contract A. According to this second form of usage, the user of a contract B can make a communication with a less cost by using the residual bandwidth of the contract A, although the communication quality is not guaranteed for the contract B. If there are a plurality of packet flows for the same contract, packet switching is carried out equally to each of the packet flows or according to the protocol and the priority of each packet flow just like in the first form of usage.

None of the conventional packet switches is provided with such the function for guaranteeing communication services for respective contracts stated above and the function for controlling transmission of a plurality of packet flows under the same contract according to the priority with respect to such variable length packets as IP packets.

FIG. **3** shows a format of the packets handled by the packet switch of the present invention.

A packet **30** comprises a header portion **31** and a data portion **32**. The header portion **31** includes packet priority information **33**, a service request **34**, protocol information **35**, a source address **36**, and a destination address **37**. The packet priority information **33** indicates the processing priority of the packet. The service request **34** indicates the information of a service requested by the packet, for example, a request for reliability and high speed performance. These items are equivalent to the priority (0 to 2nd bits), the low delay request (3rd bit), the high through-put request (4th bit), and the high reliability information (5th bit) of the TOS (Type Of Service) field defined in the header of each IP packet used for the Internet.

The protocol information **35** indicates the type of the communication protocol used for the packet, that is, the type of the packet. If the packet **30** is an IP packet, the protocol information **35** identifies the type of the transport layer header **39** following the header **31**. For the most frequently used protocols, TCP (Transmission Control Protocol), UDP (User Datagram Protocol), ICMP(Internet Control Message Protocol), and IGMP (Internet Group Management Protocol) are well known.

The source address **36** indicates the address of a source device from which the packet is transmitted. The destination address **37** indicates the address of a destination device to which the packet is forwarded. These addresses correspond to the SA (Source IP Address) and the DA (Destination IP Address) in a case of IP address system.

The IP address comprises an IP network address and an IP host address. The IP network address identifies a sub-network connected to a source or destination terminal and the IP host address identifies the source or destination terminal. The communication service contract described above can be identified with the combination of such a source IP network address and a destination IP network address.

Next, description will be made for the mapping tables **40** and **50** referred by the transmission priority control unit **5** to classify transmission packets.

FIG. **4** shows an example of the mapping table **40** referred to determine transmission priorities among a plurality of packet flows transmitted under respective contracts.

The mapping table **40** is a two dimensional matrix arranging, for example, the protocol **35** included in each packet header **31** as column information **41** and the priority information **33** included in the packet header **31** as row information **42**, wherein a value is assigned to indicate the transmission priority corresponding to each combination of the priority information **33** and the protocol **35**. Each value (0 to 7) of the transmission priority defined in the mapping table **40** also indicates the value of a second index k of the queue Qik formed in the transmission buffer area **83**-i shown in FIG. **1**.

The type of the protocol **35** may be classified into, for example, TCP and UDP according to the transport layer protocol of the OSI reference model. It may also be classified according to a layer upper than the transport layer, for example, into HTTP (Hyper Text Transfer Protocol), FTP (File Transfer Protocol), TELNET (Terminal Connection), SMTP (Simple Mail Transfer Protocol), and etc. depending on the port number if the protocol type is TCP.

Generally, in the case of an interactive type or connection oriented network protocol such as TCP, the source terminal transmits a next packet after receiving an ACK packet from the destination terminal. On the contrary, in the case of a batch type or connectionless network protocol such as UDP, the source terminal can transmit packets one after another without waiting for an ACK packet from the destination terminal. Consequently, if a higher transmission priority is given to packet flows of the interactive type or connection oriented network protocol and a lower priority is given to packet flows of the batch type or connectionless network protocol in the mapping table **40**, it is able to improve the total through-put of the packet switch **1**.

FIG. **5** shows a configuration of the mapping table **50** referred by the transmission priority control unit **5** to identify the contract associated with each of the transmission packets and to specify a queue group corresponding to the contract from among the queue groups **83-1** to **83**-j shown in FIG. **1**.

The mapping table **50** is a two-dimensional matrix arranging the source network address included in the source address **36** of each packet header **31** as column information **51** and the destination network address included in the destination address **37** of the packet header **31** as row information **52**, wherein an identification number (contract identifier) of queue group is assigned corresponding to each combination of a source network address and a destination network address.

The reason why transmission packets are buffered in queue groups by contract in the present invention is that reading of packets from the transmission buffer **83** is controlled so as to provide each transmission packet flow with a contracted bandwidth. In the embodiment shown in FIG. **5**, a contract is identified by the combination of a source network address and a destination network address, and the identification number of a queue group corresponding to each contract is defined in the mapping table **50**. Each contracted bandwidth is shown in parentheses for reference in the mapping table **50**.

A queue group identification number as described above is assumed as the value of the first index i of the queue Qik formed in the transmission buffer area **83**-i. In the illustrated example, the communication between a terminal connected to the sub-network Am and a terminal connected to the sub-network An is carried out under the contract A to which the bandwidth BW1 is assigned. It will thus be understood that packets for this contract A are queued in the queue group

83-1. For each combination of a source network address and a destination network address, which is not included in any specific contract, a null indicated with a symbol "-" is set. Packets having the address information corresponding any of such combinations are discarded by the transmission priority control unit **5**. Discarding of such packets that are communicated between non-contracted networks may contribute to improve the security of the packet communication.

The transmission priority control unit **5** refers to the two types of mapping tables **40** and **50** as described above in order to queue transmission packets in the transmission buffer **83**. Consequently, a plurality of queue groups **83-1** to **83**-j are formed in the transmission buffer **83**. Each of those queue groups **83-1** to **83**-j has an identification number defined in the mapping table **50** as shown in FIG. **1**. And, in each of those queue groups are formed a plurality of queues corresponding to a transmission priority defined in the mapping table **40** respectively as shown in FIG. **6**.

Transmission packets stored in the queue groups **83-1** to **83**-j are read out by the packet read-out circuits **82-1** to **82**-j according to the transmission priority for each queue group, then transmitted to the packet transmission unit **6** by the packet read-out circuit **81**, while the contracted bandwidth of each queue group is guaranteed.

Hereunder, the operation of the packet read-out circuit **81** will be described with reference to FIGS. **7** and **8**.

FIG. **7** shows a configuration of the management table **60** which is referred by the packet read-out circuit **81** to control bandwidths. The management table **60** comprises a plurality of records corresponding to the queue groups **83-1** to **83**-j. Each of those records comprises a queue group identification number **61**, a bandwidth **62** assigned to each queue group, and a counter area **63** for counting the length of transmission data read out from each queue group.

FIG. **8** shows a flow chart for a control operation performed by the packet read-out circuit **81**.

The packet read-out circuit **81** initializes count values in the counter area **63** of the management table **60** and the value of a parameter p to zero respectively (steps **102** and **104**), then increments the value of the parameter p (step **106**). After this, the packet read-out circuit **81** compares the value of CNT(p) of the counter **63** with the value of BW(p) of the bandwidth **62** in the p-th record of the management table **60**. If the CNT(p) is equal to or more than the BW(p), the packet read-out circuit **81** subtracts the value of BW(p) from the value of CNT(p) (step **120**), then compares the parameter p with the maximum value j (step **124**). If the value of the parameter p is less than the maximum value j, the control sequence returns to step **106**, otherwise returns to step **106** after clearing the value of parameter p to zero.

If the CNT(p) value is less than the BW(p) value in step **108**, the packet read-out circuit **81** reads out a packet from the p-th queue group QG(p) of the output buffer **83** and transmits the packet to the packet transmission unit **6** (step **110**). If there is no output packet to be read out in the queue group QG(p)(step **112**), the packet read-out circuit **81** clears the counter CNT(p) value to zero (step **122**), then performs the processing of step **124**. When the packet read-out circuit **81** reads out a packet from the queue group QG(p), the packet read-out circuit **81** adds the packet length to the counter CNT(p) (step **114**), then returns to step **108**.

FIG. **9** shows an explanatory view of a sequence of packets read out through the read control operation described above.

It is assumed here that packets are read out from three queue groups QG(1), QG(2), and QG(3) to which bandwidths BW(1), BW(2), and BW(3) are assigned, respectively, for simplifying the description. The rate of the above bandwidths is assumed to be BW(1):BW(2):BW(3)=3:2:1.

According to the control flow chart shown in FIG. **8**, the value of parameter p is changed so that each queue group can get a packet read cycle cyclically. Packets are read out from each queue group continuously until the total length of read packets exceeds the value of a contracted bandwidth BW(p) registered in the management table **60**. When the total length of the read packets exceeds the value of the contracted bandwidth BW(p), the packet read-out circuit **81** subtracts the contracted bandwidth BW(p) value from the counter CNT(p) value, then transfers the packet read cycle to next queue group.

Consequently, as shown in FIG. **9**, when packets P1, P2, and P3 are read out from the first queue group QG(1), the read cycle is transferred to the second queue group QG(2). At this time, the counter CNT(1) value of the queue group QG(1) indicates the length of the residual part of the packet P3 shown with a dotted line.

When packets P4 and P5 are read out from the second queue group QG(2), the read cycle is transferred to the third queue group QG(3). Since the contracted bandwidth of the third queue group QG(3) is small, the read cycle is transferred to the first queue group QG(1) when one packet P6 is read out in this example.

When packets P7 and P8 are read out from the first queue group QG(1), the read cycle is transferred to the second queue group QG(2). In this example, since a long packet P5 was read out from the second queue group QG(2) in the previous read cycle, the value of the counter CNT(2) exceeds the value of the contracted bandwidth BW(2). Consequently, in this read cycle, no packet is read out from the second queue group QG(2) and the read cycle is transferred to the third queue group QG(3) after performing subtraction on the value of the counter CNT(2). By repeating the read operations stated above, a series of packets are sent out to the output port OUT sequentially, in the order of P1, P2, . . . , P14. Thus, although the contracted bandwidth is exceeded temporarily, as just like outputting of packets P4 and P5 from the queue group QG(2), the contracted bandwidths of all the queue groups are guaranteed for packet switching in a long-range view.

According to the controlling of packet read-out described above, if it is detected that there is no transmission packet in any one of the queue groups, a chance to read out packets comes round to residual queue groups in a shorter cycle. Consequently, it becomes possible for residual queue groups to effectively use the bandwidth assigned for the empty queue group. In addition, if all the queue groups except one become empty, read cycles are given to the residual queue group almost continuously. As a result, the queue group can use the bandwidth of the output line all to itself, thus receiving a benefit of services over its contracted level.

In FIG. **5**, the j-th queue group is used to store such transmission packets that have an extra combination of a source network address and a destination network address other than that correspond to bandwidth guarantee contracts registered in the table **50**. For the j-th queue group, assigned is a residual bandwidth BWj which is a residual bandwidth of the output line left by the other contractors. The user of the residual bandwidth BWj shares the residual bandwidth with other non-contract users, so the user can receive a benefit of less-cost communications, although the communication service quality is not so good.

According to the packet read-out control of the present invention, when a bandwidth guaranteed contractor stops communications, the bandwidth for the contractor is released to the users of the residual bandwidth BWj. It is assumed that, as an exaggerated example, a specific user X makes a contract for guaranteeing the exclusive use of the full bandwidth of the output line **14C** and the residual bandwidth of the user X is assigned to the other users. In this case, by modifying the control flow chart shown in FIG. **8**, it is also able to realize such a packet read-out control of the head-of-line scheduling that release the full bandwidth of the line **14C** to the other users until the next packet flow of the user X arrives whenever the packet flow of the user X stops.

In the case of such a head-of-the-line scheduling sequence, packets are basically read out repetitively from a queue group of the user X. If there is no packet to be read out in the queue group of the user X, one of the other queue groups is accessed. In this case, packets may be read out one after another by checking whether there is a new packet arrived in the queue group of the user X each time one packet is read out. According to the present invention, therefore, it is possible to realize not only the first usage form, but also the second usage form, thereby providing communication services appropriately to both of the bandwidth contractor and other non-contractors.

The read-out circuit **82** controls the order of packets to be read out from each queue group, for example, the order of the packets P1, P2, P3, P7, . . . in the first queue group QG(1). In other words, the contracted bandwidth BW(i) assigned to each queue group is further divided by the read-out circuit **82** so as to be assigned to a plurality of queues Qi to Qin in the queue group.

In this case, how to assign a bandwidth to each queue is varied widely. In the most basic method, packets are read out from the queues in the order of descending priority and subsequent priority queue is accessed to read out packets when the higher priority queue becomes empty. Instead of this method, it is also possible to assign a bandwidth to each queue according to the order of priority beforehand, then give a different read-out time to each queue just like the method used by the read-out circuit **81**. In this case, the controlling procedure is modified so that the queue group QG(p) is replaced with the queue Q(p) in the flow chart shown in FIG. **8**.

In order to simplify controlling of packet reading more, it is also possible to assign the number of packets readable continuously for each queue beforehand regardless of the packet length so that packets are read out from the next queue when the assigned number of packets are read out from one queue according to the order of descending priority. It is also possible to read out packets one by one from each queue simply according to the order of descending priority, if necessary.

According to the flaw of output packets from the output buffer **83** to the output port OUT, the read-out circuit **82** is located in front of the read-out circuit **81**. However, in a viewpoint of the read-out control procedure, a queue group is determined first by the read-out circuit **81**, then the read-out circuit **82** corresponding to the determined queue group reads out packets from one of the queues in the group as shown in the flow chart in FIG. **8**.

In the packet switch shown in FIG. **1**, a read-out circuit **82** is disposed for each queue group in order to make it easier to understand the present invention. In this case, while packets are read out from one queue group and passed to the next stage read-out circuit **81**, none of the read-out circuits **82** in other queue groups transmits packets. Consequently, all the packets stored in the output buffer **83** may be read out by one read-out circuit **82** by switching over the queue groups in turn, in an actual application.

In addition, if an address table for indicating the addresses of the first packets of the respective queues is prepared for each queue group and if the address table is referred in accordance with the queue group QG(p) specified in step **110** in FIG. **8** to read out packets from a queue identified by the priority, the functions of both read-out circuits **81** and **82** can be realized by a processor or a dedicated hardware circuit.

FIG. **10** indicates the packet read-out circuit **82** in another embodiment.

This packet read-out circuit **82** comprises a complete priority type controlling circuit **821** for controlling the read out of packets so that all the packets are read out from the first priority queue, then from the next and subsequent priority queues sequentially, a rate based scheduler type of controlling circuit **822** for controlling the reading out of packets so that packets are read out from a plurality of queues in a queue group according to each assigned bandwidth, and a selector circuit **823** for selecting any one of the two controlling circuits according to a control signal from the management unit **9**. One of the controlling circuits **821** and **822** is activated according to the control signal from the management unit **9**. According to this embodiment, the read-out mode of the transmission packet in the packet switch **1** can be switched over in a moment by a command supplied from a management terminal operated by the network manager.

In the above embodiment, description has been made for packet switching in the network layer of the OSI reference model represented by IP packets, but the technical concept of the present invention may also apply to the data link layer of the OSI reference model, thereby controlling the bandwidth for each of the queue groups described above, for example, in an ETHERNET LAN switch.

In the embodiment described with reference to FIG. **5**, the bandwidth of each queue group is controlled under each contract identified from both source and destination network addresses. The present invention may also apply to another embodiment in which the transmission priority control unit **5** classifies transmission packets by protocol, for example, TCP and other protocols into two groups, and transmission packets are queued into a plurality of queues for each queue group according to the packet priority included in the header information of each packet. In this case, bandwidths may be assigned to queue groups in such a manner that the whole bandwidth of an output line is assigned only to TCP packet flows and the residual bandwidth is assigned to the packet flows of other protocols.

As to be understood from the above description, according to the present invention, it is possible to provide a packet switching in which the packet transmission priority is changed for each network connected to the node apparatus (packet switch) in a communication network such as the Internet in which a plurality of networks are connected through a common line. In addition, according to the present invention, it is also possible to provide a packet switch and a packet switching method that can change the transmission priority even with a communication protocol as needed while the communication bandwidth is guaranteed for each user according to the communication service contract.

What is claimed is:

1. A packet switch, comprising:

a plurality of input port interfaces;

a plurality of output port interfaces, and a packet relaying unit for switching variable length packets received from said input port interfaces to one of said output port interfaces corresponding to the destination address of each of said packets,

wherein each of said output port interfaces comprises:

a buffer memory for storing transmission packets,

a transmission priority controller for classifying transmission packets, passed from said packet relaying unit into a plurality of queue groups to which specific bandwidths are assigned respectively, according to the header information of each of said transmission packets and for queuing said transmission packets in said buffer memory so as to form a plurality of queues according to transmission priority in each queue group,

a packet read-out controller for reading out transmission packets according to the transmission priority while guaranteeing the bandwidth assigned to each of said queue groups, and

a packet transmission circuit for transmitting said transmission packets read out by said packet read-out controller to an output port associated with the output port interface,

wherein said transmission priority controller includes a first management table for defining an identifier for each of said queue groups corresponding to a combination of a source network address and a destination network address and a second management table for defining the transmission priority corresponding to a combination of a communication protocol and a packet priority, and

wherein said transmission priority controller identifies a queue group corresponding to each of said transmission packets by referring to said first management table based on the source and destination network addresses included in the header of the transmission packet and identifies the transmission priority corresponding to the transmission packet by referring to said second management table based on the, communication protocol information and packet priority information included in the header of the transmission packet.

2. A packet switch according to claim **1**, wherein said packet read-out controller comprises:

a management table for defining a threshold value in proportion to said bandwidth corresponding to each of said queue groups; and

control means for accessing a plurality of said queue groups cyclically to read out transmission packets continuously from each of said queue groups until the total length of read out packets exceeds the threshold value defined in said management table, and for storing a surplus length of a packet read out over the threshold value as an initial value for counting the total length of packets in the next cycle to read out packets from the queue group.

3. A packet switch according to claim **2**, wherein said control means includes means for checking, before reading transmission packets from each of said queue groups, the initial value for counting the total length of packets associated with each of said queue groups, subtracting the threshold value from the initial value if the initial value of one

queue group exceeds the threshold value, and switching a queue group to be an object for reading out transmission packets therefrom, from said one queue group to a next queue group without reading transmission packets from said one queue group.

4. A packet switch, comprising:

a plurality of input port interfaces;

a plurality of output port interfaces, and a packet relaying unit for switching variable length packets received from said input port interfaces to one of said output port interfaces corresponding to the destination address of each of said packets,

wherein each of said output port interfaces comprises:

a buffer memory for storing transmission packets,

a transmission priority controller for classifying transmission packets, passed from said packet relaying unit into a plurality of queue groups to which specific bandwidths are assigned respectively, according to the header information of each of said transmission packets and for queuing said transmission packets in said buffer memory so as to form a plurality of queues according to transmission priority in each of said queue groups,

a packet read-out controller for reading out transmission packets according to the transmission priority while guaranteeing the bandwidth assigned to each of said queue groups, and

a packet transmission circuit for transmitting said transmission packets read out by said packet read-out controller to an output port associated with the output port interface,

wherein said transmission priority controller specifies, based on the protocol information included in the header of each of transmission packets passed from said packet relaying unit, the queue group corresponding to the transmission packets, whereby each of said transmission packets is queued in one of the queues in the specified queue group.

5. A packet switch according to claim 4, wherein said packet read-out controller comprises:

a management table for defining a threshold value in proportion to said bandwidth corresponding to each of said queue groups; and

control means for accessing a plurality of said queue groups cyclically to read out transmission packets continuously from each of said queue groups until the total length of read out packets exceeds the threshold value defined in said management table, and for storing a surplus length of a packet read out over the threshold value as an initial value for counting the total length of packets in the next cycle to read out packets from each of said queue groups.

6. A packet switch according to claim 4, wherein said control means includes means for checking, before reading transmission packets from each of said queue groups, the initial value for counting the total length of packets associated with each of said queue groups, subtracting the threshold value from the initial value if the initial value of one queue group exceeds the threshold value, and switching a queue group to be an object for reading out transmission packets therefrom, from said one queue group to a next queue of group without reading transmission packets from said one queue group.

* * * * *